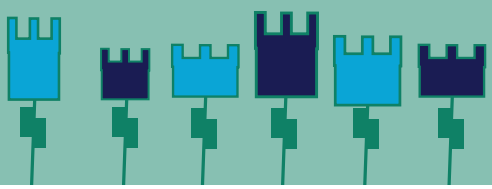
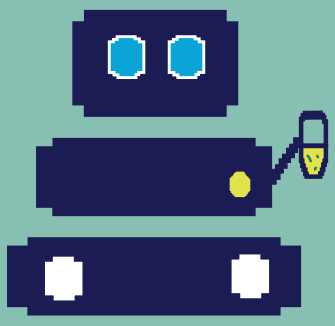
APSCE CTE-STEM 2022

6TH APSCE INTERNATIONAL CONFERENCE ON COMPUTATIONAL THINKING AND STEM EDUCATION

CONFERENCE PROCEEDINGS
15TH - 17TH JUNE 2022





ORGANISED BY:

Asia-Pacific Society for Computers in Education

HOSTED BY:

Centre for Education and Learning
Leiden-Delft-Erasmus Universities

TU Delft

SUPPORTED BY:

EATEL

ONDERWIJS NETWERK ZUID-HOLLAND

BETA

zhaw Life Sciences and Facility Management
Institute of Computational Life Sciences

Open Universiteit

Contents

0_paper 1

2022_2_book-colophon-authors_V1.1 (Feedbakc from NardieFrederique)_20220601 5

2022_3_preface_20220608_V1 10

Topics addressed in this volume 11

2022_4_book-keynote abstract and title_v1.2 (Feedback from MarliesFrederique)_20220607 13

2022_5_toc(Table of contents) 16

1_paper_21 19

2_paper_11 24

Untitled 24

3_paper_14 30

4_paper_30 36

5_paper_12 40

1.INTRODUCTION 40

2.METHOD 41

2.1.Participants 41

2.2.Lesson Design 41

2.3.Measurements 42

2.4.Procedure 42

3.RESULT 42

3.1.RQ1: Effect on CT Cognitive Performance 42

3.2.RQ2: Effect on Coding Attitudes 42

4.DISCUSSION AND CONCLUSION 43

5.REFERENCES 43

6_paper_6 44

7_paper_10 50

Contents (continued)

8_paper_24 56
1. INTRODUCTION 56
2. BACKGROUND 56
3. METHOD 57
3.1. Respondents 57
3.2. Survey 57
4. FINDINGS 58
5. DISCUSSION 59
5.1. Addition of a “mixed-disciplines” category 59
5.2. Conceptions of CT 59
6. LIMITATIONS 59
7. CONCLUSION 59
8. REFERENCES 60
9. APPENDIX 61
9_paper_33 62
10_paper_2 64
11_paper_18 70
12_paper_41 76
13_paper_3 82
14_paper_38 86
15_paper_4 90
16_paper_27 95
17_paper_34 101
18_paper_29 104
19_paper_26 106

Contents (continued)

20_paper_35 108
21_paper_1 110
22_paper_25 116
23_paper_40 122
24_paper_43 128
25_paper_7 135
1. ELEMENTARY CT INTEGRATION 135
2. CT INTEGRATION APPROACHES 135
3. RUBRIC DEVELOPMENT PROCESS 136
4. TYPES OF INTEGRATION 136
5. QUALITY INDICATORS 136
5.1 Conceptual Coherence 137
5.2 Role of Computational Technology 137
5.3 Assessment 137
5.4 Multiple Representations 138
5.5 Play 138
5.6 Equity 138
6 RUBRIC SCALE AND APPLICATION 139
7 CONCLUSION 139
8 REFERENCES 139
26_paper_9 141
27_paper_15 146
0_paper 152

CTE-STEM 2022 | Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022

15-17 June 2022, Delft, The Netherlands

Editors
Xiaoling Zhang[1], Christian Glahn[2], Nardie Fanchamps[3], Marcus Specht[4]

[1] Delft University of Technology	x.zhang-14@tudelft.nl	orcid: 0000-0003-0951-0771
[2] Zurich University of Applied Sciences	christian.glahn@zhaw.ch	orcid: 0000-0002-2701-3579
[3] Open Universiteit Nederland	nardie.fanchamps@ou.nl	orcid: 0000-0001-7509-2251
[4] Delft University of Technology	m.m.specht@tudelft.nl	orcid: 0000-0002-6086-8480

Marcus Specht and Xiaoling Zhang are responsible for proceedings publication communication and management. Christian Glahn and Nardie Fanchamps are responsible for organizing and communicating teacher-related activities.

Keywords
Computational Thinking, STEM (Science, Technology, Engineering, Mathematics), Education, Games, Teachers, Programming, Reviews, Concepts

Citation:
Zhang, X. Glahn, C., Fanchamps, N., & Specht, M. (Eds.). (2022). *Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022*. CTE-STEM. TU Delft Open Publishing.

Published by
TU Delft OPEN Publishing | Delft University of Technology, The Netherlands

ISBN: 978-94-6366-563-6
ISSN: 2664-5661
DOI: https://doi.org/10.34641/mg.37

Copyright statement

This work is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence
© 2022 published by TU Delft OPEN Publishingon behalf of the authors

Electronic version of this book is available at:
https://proceedings.open.tudelft.nl/cte-stem2022

Cover design made by Marlies Peter

Copyright clearance made by the TU Delft Library copyright team

Fundings
Thanks to the following organizations for their support to make this event happen: Leiden-Delft-Erasmus Center for Education and Learning (LDE-CEL), TU Delft, EATEL, Bètasteunpunt Zuid-Holland, ZHAW and Open Universiteit.

Disclaimer
Every attempt has been made to ensure the correct source of images and other potentially copyrighted material was ascertained, and that all materials included in this book have been attributed and used according to their license. If you believe that a portion of the material infringes someone else's copyright, please contact the editors of this book.

Acknowledgements
We greatly appreciate Marlies Petter, Sylvia Walsarie Wolff, all reviewers and people from 4TU.CEE and LDE-CEL (Leiden-Delft-Erasmus Centre for Education and Learning) for their support for organising this conference.

TU Delft OPEN

Review Process

All work included in this proceedings were double blind peer-reviewed.

List of Reviewers

Ankur Singh Bist	Govind ballabh pant university of agri. and tech
Ann Ming Lui	Hong Kong Baptist University
Arnon Hershkovitz	Tel Aviv University
Baichang Zhong	Nanjing Normal University
Beng Keat Liew	Republic Polytechnic
Bin Haw Chang	Ngee Ann Polytechnic
Bin Lu	California State University Sacramento
Chee-Kit Looi	National Institute of Education
Christian Glahn	Zurich University of Applied Sciences
Chronis Kynigos	Εργαστήριο Εκπαιδευτικής Τεχνολογίας, Πανεπιστήμιο Αθηνών
Daren Ler	National University of Singapore
David Weintrop	University of Maryland
David Zeigler	California State University, Sacramento
Edna Chan	Singapore Polytechnic
Efthimia Aivaloglou	University of Leiden
Estefanía Martin-Barosso	Universidad Rey Juan Carlos
Evan Patton	Massachusetts Institute of Technology
Faron Moller	Swansea University
Florence Sullivan	UMass, Amherst
Fredrik Heintz	Linköping University
Giora Alexandron	Weizmann Institute of Science
Gregorio Robles	Universidad Rey Juan Carlos
Guang Chen	Beijing Normal University
Gwo-Jen Hwang	National Taiwan University of Science and Technology
Heinz Ulrich Hoppe	University Duisburg-Essen / RIAS Institute Duisburg
Hillary Swanson	Utah State University
Hui-Chun Hung	National Central University
Hüseyin Özçınar	PAU
Hyo-jeong So	Ewha Womans University
Ibrahim H. Yeter	Nanyang Technological University
Irwin Kuo-Chin King	The Chinese University of Hong Kong
Ivica Boticki	FER UNIZG
James H. Davenport	University of Bath
Jan Vahrenhold	Department of Computer Science, Westfälische Wilhelms-Universität Münster
Jinbao Zhang	Beijing Normal University
Johan Jeuring	Utrecht University
Jon Mason	Charles Darwin University
Joshua Ho	The University of Hong Kong

Ju-Ling Shih	National Central University
Junfeng Yang	Hangzhou Normal University
Junjie Shang	Peking University
Ki-SangSong	Korea National University of Education
Kok Cheng Tan	Republic Polytechnic
Lakshmi Dhandabani	Adithya Institute of Technology
Lanqin Zheng	Beijing Normal University
Li-Chieh Chang	National Central University
Linda Wai-Ying Kwok	The Education University of Hong Kong, Hong Kong
Marc Jansen	University of Applied Sciences Ruhr West
Marcos Román-González	UNED
Marcus Specht	Delft University of Technology
Maria Marcelino	University of Coimbra
María Zapata-Cáceres	Universidad Rey Juan Carlos
Mi Song Kim	Western University
Ming Lai	The Education University of Hong Kong
Ming-Puu Chen	National Taiwan Normal University
Morris Siu-Yung Jong	The Chinese University of Hong Kong
Nardie Fanchamps	Open University The Netherlands
Niels Pinkwart	Humboldt-Universität zu Berlin
Nuno Otero	Linnæus University
Oka Kurniawan	Singapore University of Technology and Design
Paul Hennissen	Zuyd University of applied sciences
Peter Seow	National Institute of Education
Pierre Gorissen	Hogeschool van Arnhem en Nijmegen. iXperium / Centre of Expertise Leren met ict
Roland Klemke	Open University of the Netherlands
Sabiha Yeni	Leiden University
Samuel Chi-Cheng Chang	National Taiwan Normal University
Shao-Chen Chang	National Taiwan Normal University, NTNU
Shu-Hsien Huang	National Chin-Yi University of Technology
Sridhar Iyer	IIT Bombay
Sungwan Han	Gyeong-in National University of Education
Teresa Cerratto-Pargman	Stockholm University
Tosti H. C. Chiang	National Taiwan Normal University
Valentina Dagiene	Vilnius university Institute of Mathematics
Vijay Raisinghani	NMIMS
Wei Cheng	Nanjing University of Posts and Telecommunications
Xiaoling Zhang	Delft University of Technology
Xu Li	University of Arizona South
Ying Hwa Kee	Nanyang Technological University

Ying-Tien Wu	National Central University, Taiwan
Yogendra Pal	NIIT University
Yuen-Tak Yu	City University of Hong Kong

Local Organizing Committee

Dr. Nardie Fanchamps	Assistant Professor Educational Sciences Open University NL
Sylvia Walsarie Wolff	Project Manager Centre for Education and Learning
Xiaoling Zhang	PhD Computational Thinking Education
Marlies Petter	Communication Officer

Welcome from the APSCE SIG on Computational Thinking Education and STEM Education

Preface

The 6th APSCE International Conference on Computational Thinking and STEM Education 2022 (CTE-STEM 2022) is organized by the Asia-Pacific Society for Computers in Education (APSCE) and hosted by the Leiden-Delft-Erasmus Centre for Education and Learning (LDE-CEL). CTE-STEM 2022 is hosted for the first time in Europe by the Delft University of Technology (TU Delft), Delft, the Netherlands. This conference continues from the success of the previous four international Computational Thinking conferences organized by the National Institute of Education and Nanyang Technological University (NIE/NTU). This conference invites CT as well as STEM researchers and practitioners to share their findings, processes, and outcomes in the context of computing education or computational thinking.

As research shows the topic of Computational Thinking needs further clarification, embedding, practices. CT links to many highly relevant and important topics as programming education, data science and Artificial Intelligence applied in different educational and professional settings. Understanding the underlying algorithms of data and machine learning driven solutions, working on structured problem solving and scalability as also using big data and data sciences in diverse domains are key skills for future generations. To understand what skills future generations need and how we can train them to learn and apply these skills for solving problems relevant for society are of highest importance. This applies to issues of using computational tools in engineering, social sciences, management, and many other domains.

CTE-STEM 2022 is a forum for worldwide sharing of ideas as well as dissemination of findings and outcomes on the implementation of computational thinking and STEM development. The conference comprises keynote speeches, workshops, and paper presentations and therefore brings together researchers, innovators, and professionals. The International Teachers Forum is organized for teaching practitioners to share their practices in teaching Computational Thinking, Computing and STEM in the classroom. We believe bringing all these would create enriching experiences for educators and researchers to share, learn and innovate approaches to learning through Computational Thinking and STEM education. This year, teachers can participate in Lightning Talks to share ideas about teaching and learning CT.

Topics addressed in this volume

There are 27 papers in total, with 4 teacher forum manuscripts and 23 scientific contributions with 4 short submissions and 19 full research papers. 6 main themes are clustered in these proceedings.

Using and developing games and gamification approaches for CT and STEM education to embed learning in simplified or authentic contexts which enables focused learning and easy transfer as also fosters high motivation for learners (Chapter 1: Games in CT). Moreover, integrating and

combining STEM education and computational tooling are the focus of the subsequent chapter (Chapter 2: STEM Meets CT)

What are the challenges of educators for teaching and learning with computational tools and how can we embed CT in Schools and link it to the core subjects (Chapter 3: Teachers and CT, Chapter 4: CT and Programming in Schools)?

Analyzing the effectiveness and understanding where the field stands is addressed in Chapter 5: Concepts and Reviews. Furthermore, assessing CT is at the core of building a skills model and measuring growth and skillfulness in a continuous, reliable, and valid way. A variety of models have been developed and are discussed in Chapter 6: Design, Assessment and Evaluation of CT in Formal and Non-formal Settings.

We are very happy to have 4 keynote speakers for the conference including **Dr. Georgi Dimitrov** head of unit Digital Education, European commission talking about the role of digital skills in future Europe, **Jens Mönig** from SAP talking about his innovations in programming and SNAP! **Prof. Matti Tedre** from the University of Eastern Finland talking about the role of machine learning and our understanding of the digital world and **Prof. Maarten de Laat** from the Centre for Change and Complexity in Learning, University of South Australia on his work on hybrid AI-human problem solving in the classroom.

On behalf of APSCE and the Conference Organizing Committee, we would like to express our gratitude to all speakers as well as paper presenters for their contribution to the success of CTE-STEM 2022.

We sincerely hope everyone enjoys and gets inspired from CTE-STEM 2022.

With Best Wishes,

Professor Marcus Specht
Conference Chair, CTE-STEM 2022
Delft University of Technology (TU Delft), the Netherland

Dr. Christian Glahn
Conference Co-Chair, CTE-STEM 2022
Zurich University of Applied Sciences (ZHAW), Switzerland

Dr. Nardie Fanchamps
Conference Co-Chair, CTE-STEM 2022
Open University, the Netherlands

MSc. Xiaoling Zhang
Conference Co-Chair, CTE-STEM 2022
Delft University of Technology (TU Delft), the Netherland

Keynotes

KEYNOTE TOPIC: DIGITAL SKILLS

GEORGI DIMITROV *HEAD OF UNIT DIGITAL EDUCATION, EUROPEAN COMMISSION, DIRECTORATE GENERAL EDUCATION AND CULTURE*

ABOUT GEORGI DIMITROV

Georgi Dimitrov is responsible for the Digital Education unit in the European Commission, Directorate General for Education and Culture. He joined the European Commission in 2008 and was first involved in various roles in setting up the European Institute of Innovation and Technology (EIT). He then helped to develop and launch HEInnovate, an initiative by the European Commission and the OECD aimed at supporting universities to become more entrepreneurial. He led the development of the first Digital Education Action Plan adopted in January 2018 and also of the new Digital Education Action Plan 2021-2027 that was adopted in September 2020. Before joining the Commission, Georgi worked for a leading multinational telecommunication company and in a software start-up in Germany. Georgi studied at the University of Bonn (M.A.), the University of Erlangen-Nürnberg (PhD) and the Open University UK (MBA in Technology Management).

KEYNOTE TITLE: PROGRAMMING AS A MEDIUM

JENS MÖNIG *RESEARCHER AT SYSTEMS, APPLICATIONS & PRODUCTS IN DATA PROCESSING (SAP)*

KEYNOTE ABSTRACT:

Computers, apps and programming languages are still commonly referred to as tools that help us accomplish tasks by amplifying particular skills such as calculating and remembering. Yet as computers and their apps have evolved into channels of communication among us and our appliances, programming languages are becoming a medium letting us interface with the world and express our ideas. I will present the Snap! visual programming language and discuss its design principles from the perspective of encouraging learners to approach programming not just as a tool for production but as a medium for exploration.

Snap! Is a Scratch-like programming language that treats code-blocks as first class citizens instead of confining them to an editing modality. Embracing nested data structures and higher order functions Snap! let learners create arbitrary control structures and even custom

programming languages with just blocks. Snap! has been developed for UC Berkeley's introductory computer science course named "The Beauty and Joy of Computing".

ABOUT JENS MÖNIG

Jens Mönig is a researcher at SAP and makes interactive programming environments. He is fanatical about visual coding blocks. Jens is the architect and lead programmer, together with Brian Harvey, of UC Berkeley's "Snap! Build Your Own Blocks" programming language, used in the introductory "Beauty and Joy of Computing" curriculum. Previously Jens has worked under Alan Kay on the GP programming language together with John Maloney and Yoshiki Ohshima, helped develop Scratch for the MIT Media Lab and written enterprise software at MioSoft. Jens is a fully qualified lawyer in Germany and has been an attorney, corporate counsel and lecturer for many years before rediscovering his love for programming through Scratch and Squeak. For leisure Jens likes guitar picking and strumming his mandolin.

KEYNOTE TITLE: FROM RULE-DRIVEN TO DATA-DRIVEN COMPUTING EDUCATION IN K-12

MATTI TEDRE *PROFESSOR AT SCHOOL OF COMPUTING, UNIVERSITY OF EASTERN FINLAND*

KEYNOTE ABSTRACT:

The popular approaches to K-12 computing education today are based on analyzing and describing problems in a way that enables their solutions to be formulated as series of computational steps. Rule-based "classical" programming paradigms have come to dominate K-12 programming education, with some of their relevant key concepts and skills described under the title computational thinking (CT).

In the 2000s a number of data-driven technologies, most prominently machine learning (ML), have become commonplace in apps, tools, and services. Understanding some key ideas related to ML is becoming crucial for understanding how many key elements of our digital environment work. The power of traditional, rule-based computational thinking (CT1.0) to explain ML-driven systems is, however, limited, and new approaches to computing education are needed. A body of literature on how to teach some principles of ML and data-driven computing in K-12 education is emerging, but that body of literature relies on a set of concepts and skills very different from traditional CT1.0. This talk outlines the key changes in the conceptual landscape, educational practice, and technology for ML-enhanced CT (CT2.0) and compares it to the dominant computing education paradigm.

KEYNOTE TITLE: AI IN THE CLASSROOM – STUDENTS COLLABORATING WITH AI TO SOLVE COMPLEX PROBLEMS

MAARTEN DE LAAT *PROFESSOR AT ENTRE FOR CHANGE AND COMPLEXITY IN LEARNING, UNIVERSITY OF SOUTH AUSTRALIA*

KEYNOTE ABSTRACT:

Applications of artificial intelligence (AI) are set to transform society, including how people work and learn. This growing ubiquity of AI in society poses significant challenges for educational systems: what will citizens in the 21st century need to know about, and do with AI? Currently there is very little research and experience on how schools and teachers adopt AI into the classroom and how our students work and learn together with AI.

In this keynote I will present some current work at our Centre for Change and Complexity in Learning to help address this issue. I will showcase some initiatives where students will work together with AI to solve complex problems. Our mission is to offer an AI learning environment where students can take ownership over AI, experiment with it and develop AI to follow their imagination. The environment is a social space for exploration and critical evaluation, it's safe and inspiring. This is how we want students to treat AI. Rather than that AI is done to you, students should be able to play with AI, and through play, shape it so that AI starts to work for you and help you to go beyond your own capability.

Table of Contents

Chapter 1: Games in CT

How to Teach Coding through Stories in Early Childhood Classrooms 1
Burcu Çabuk, Gülgün Afacan Adanir and Yasemin Gülbahar

Bebras in the Digital Game < Captain Bebras > for Students' Computational Thinking Abilities 6
Yan-Ming Chen and Ju-Ling Shih

A robotic-based approach for CT development: challenges of teaching programming concepts to children and the potential of informal learning. 12
Rafael Zerega, Ali Hamidi, Sepideh Tavajoh and Marcelo Milrad

Modelling Zombies and Other Diseases 18
Stewart Powell, Faron Moller, Daniel Archambault, Phoebe Asplin, Graham McNeill, Max Sondag and Cagatay Turkay

Integrating Game-based Learning into Computational Thinking Class for Lower Primary Students: Lesson Design and Course Effect 22
Shuhan Zhang, Gary K. W. Wong and Peter C.F. Chan

Chapter 2: STEM Meets CT

Integrating CT into Biology: Using Decision Tree Models to Classify Cell Types 26
Jacqueline Nijenhuis-Voogt, Sabiha Yeni and Erik Barendsen

A STEM-based Learning Activity Instructional Design of Quadruped Bionic Robots 32
Shaun-Wen Chen and Ju-Ling Shih

Comparison of STEM, non-STEM, and Mixed-Discipline Pre-service Teachers' Early Conceptions about Computational Thinking 38
Wendy Huang, Chee-Kit Looi and Ibrahim H. Yeter

The Effect of Unplugged Programming and Visual Programming on Computational Thinking in Children Aged 5 to 7 44
Lisa Bosgoed and Nardie Fanchamps (Teacher Forum)

Chapter 3: Teachers and CT

Understanding Teachers' Attitudes and Self-Assessment Towards Computational Thinking 46
María Zapata-Cáceres, Nardie Fanchamps, Ibrahim H. Yeter, Pedro Marcelino and Estefanía Martín-Barroso

Digital Competence & Computational Thinking for Preschool Pre-service Teachers: From Lab to Practice 52
Ali Hamidi, Rafael Zerega, Sepideh Tavajoh, Marcelo Milrad and Italo Masiello

Computational Thinking in Flanders' Compulsory Education 58
Natacha Gesquière and Francis Wyffels

The TACTIDE EU project: TeAching Computational Thinklng with Digital dEvices 64
Marc Jansen, Nardie Fanchamps, Marcelo Milrad, Marcus Specht and Ali Hamidi

How the Pre-service Teachers Associate Computational Thinking with Practices of Programming? A Case Study of an Introductory Programming Course in Teacher Education 68
Megumi Iwata, Jari Laru and Kati Mäkitalo

Chapter 4: CT and Programming in Schools

Pedagogical Use of Scratch Coding for Co-Developing English Language "Locations and Directions" Building Blocks and Computational Thinking 72
Siu Cheung Kong and Wai Ying Kwok

Computational Thinking Dashboard: For learners in Jupyter notebooks 77
Bhoomika Agarwal

Solving Domain-Specific Problems with Computational Thinking 83
Sharon Calor, Izaak Dekker, Dorrith Pennink and Bert Bredeweg

Precoding skills - Teaching computational thinking to preschoolers in Singapore using unplugged activities 86
Vidhi Singhal (Teacher Forum)

Computational Thinking in Language Arts When Teaching Creative and Expository Writing 88
Aysegul Bayraktar and Yasemin Gulbahar (Teacher Forum)

Exploring Embedded Computational Thinking in STEM Teacher Education 90
Dorrith Pennink, Izaak Dekker, Sharon Calor, Bert Bredeweg and Monique Pijls (Teacher Forum)

Chapter 5: Concepts and Reviews

An Exploratory Study of the Relationship between Computational Thinking and Creative Attitudes among University Students 92
Masanori Fukui, Yuji Sasaki and Tsukasa Hirashima

A Review of Reviews on Computational Thinking Assessment in Higher Education 98
Xiaoling Zhang and Marcus Specht

Developing a Continuous, Rather Than Binary, Classification for Measuring STEM Jobs .. 104
Ted Carmichael, John Stamper and John Carney

Computational Thinking, History and Non-formal learning- A well-crafted blend! 110
Irene Silveira Almeida and Ajita Deshmukh

Chapter 6: Designing, Assessment, and Evaluation of CT in Formal and Non-formal Settings

Design of an Evaluative Rubric for CT Integrated Curriculum in the Elementary Grades .. 117
Florence Sullivan, Lian Duan and Emrah Pektas

Scaffolded programming projects to promote computational thinking 123
Victor Koleszar, Alar Urruticoechea, Andrés Oliveri, María del Rosario Schunk and Graciela Oyhenard

Log-Based Multidimensional Measurement of CT Acquisition 128
Rotem Israel-Fishelson and Arnon Hershkovitz

DOI: 10.34641/ctestem.2022.449

How to Teach Coding through Stories in Early Childhood Classrooms

Burcu ÇABUK (cabuk@education.ankara.edu.tr, Ankara University), Gülgün AFACAN ADANIR (gafacan@ankara.edu.tr, Ankara University), Yasemin GÜLBAHAR(gulbahar@ankara.edu.tr, Ankara University)

ABSTRACT

Computational thinking is important for everyone and focuses on solving problems, designing systems, and understanding human behavior through fundamental concepts of computer science. Early years are important for young students to learn coding, and at the same time, they can improve problem solving and computational thinking skills. Coding can be introduced to students through unplugged and plugged activities. Unplugged activities are more appropriate for young students since they contain concrete practices and teach main coding concepts in an entertaining, motivating, and challenging way in accordance with the developmental levels of children. Owing to this fact, the purpose of the current study was to demonstrate the implementation of stories as unplugged activities for teaching coding at an early childhood level. In the context of this study, preschool teacher candidates were considered and a 14-weeks-training (including theory and practice sessions) was implemented to teach computational thinking, coding concepts, and STEAM activities. After this training, teacher candidates engaged in creating unplugged activities to teach coding to preschoolers. In this respect, the study considered two different unplugged activities: Storigami (implementation of origami activities through stories) and Coding through Stories. Hence, 15 teacher candidates learned successfully how to teach coding and created various stories to teach coding to preschoolers. This paper introduces these activities as appropriate unplugged activities on the way of introducing coding concepts to young children. In the light of the findings, suggestions were presented to preschool teacher candidates, teachers, teacher training instructors and researchers.

KEYWORDS

coding, early childhood, unplugged, story based learning

1. INTRODUCTION

Computational thinking was identified as a major skill for every person, not only for computer scientists. The concept of computational thinking was firstly proposed by Wing in 2006. According to Wing, computational thinking covers "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006, p.33).

Computational thinking was proposed as a focused approach for problem solving, and at the same time integrating thought processes that employ abstraction, decomposition, algorithmic design, evaluation and generalizations (Selby & Woollard, 2013). Abstraction refers to defining a problem or situation while focusing on information needed to solve the problem; decomposition means breaking data, operations or problems into smaller pieces; algorithmic design refers to designing the steps necessary to solve any problem; evaluation and generalizations involve evaluation of the solution and its generalization to subject domain.

Prior research has demonstrated that preschool children can build and program simple robotics projects (Wyeth, 2008) as well as they can learn ideas from engineering and computer programming while building their computational thinking skills (Bers, 2008). Computational thinking helps children develop fine-motor skills and hand-eye coordination while engaging in collaboration with other children and learn to work in teams. Furthermore, it allows preschool teachers to integrate academic content with the creation of meaningful products in a fun and playful technique (Resnick, 2003).

2. LITERATURE REVIEW

Digital literacy is seen as one of the essential skills of the twenty-first century, and provides students with gaining digital skills and learning coding (Judge, Puckett & Çabuk, 2004). Coding is defined as "the process of creating step-by-step instructions a computer understands and needs in order for its programs to work" (McLennan, 2017, p.1). Early years are important for young students to learn coding, as well as they can improve problem solving skills and computational thinking (Lee & Junoh, 2019).

Coding can be introduced to students through unplugged and plugged activities. Unplugged activities are "a widely used collection of activities and ideas to engage a variety of audiences with great ideas from Computer Science, without having to learn programming or even use a digital device" (Bell & Vahrenhold, 2018, p. 497). While teaching coding to young students, unplugged activities should be initially employed (Lee & Junoh, 2019) since they are more developmentally appropriate for young students and contain concrete practices and teach main coding concepts in an entertaining, motivating, and challenging way (Battal et al., 2021).

At early childhood level, it is important to introduce directional words (e.g. move forward, move backward, turn left, turn right) and sequential words (e.g. first, second, third) before starting any coding activity (Lee, 2020). After this introduction, learners can be provided with a grid-paper activity to understand the movements through

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

directional and sequential words. Finally, for the developmental needs of young children, the most essential activity to reinforce these coding skills is a movement game that also supports the children's gross motor skills.

Unplugged activities are more appropriate for the age group 2-7 and can be provided without using digital devices (Saxena et al., 2020). Unplugged activities are based on various methods like games, teamwork, tricks and employ various objects like cards, boards and stickers (Nishida et al., 2009). Stories are one of the methods used while implementing unplugged activities at the early childhood level. Stories can be employed for demonstrating events by time sequence (Lee & Junoh, 2019).

Picture storybooks have an indisputable importance for the progress of children especially in cognitive, social-emotional and language development and for supporting their education in early childhood (Deniz & Gönen, 2020). Stories are also used as one of the dynamic assessment methods. Creating a story by looking at a picture book or other materials and talking about personal experiences related with the written materials are some of the good practices used in early childhood settings (Işıtan & Turan, 2014).

Storigami technique, known as storytelling accompanied by origami, which is the art of paper folding, has great benefits in terms of reflecting the multifaceted development, change and difference seen in early childhood to the child and his/her life (Tanju Aslışen, 2021). While doing storigami activities, creativity, aesthetic perception, sense of success, focusing attention, following the model, seeing different perspectives, part-whole relationship, cooperation, obeying the rules, problem solving, expressing feelings and thoughts artistically, self-confidence, hand and finger muscles, expressing oneself verbally, three-dimensional thinking and etc. are developed (Tuğrul & Kavici, 2002).

3. METHODOLOGY

3.1. Procedure

In the context of this study, preschool teacher candidates were considered and a 14-weeks-training (including theory and practice sessions) was implemented to teach computational thinking, coding concepts, and STEAM activities. In each week, teacher candidates are involved in online training sessions. Each session takes approximately two hours and covers theoretical and practical concepts. After each session, teacher candidates were required to create related activities to be used in early childhood classrooms.

One session of the training covers the concepts of Storigami and Coding through stories. After this session, teacher candidates engaged in creating activities related to Storigami and Coding through stories. Each candidate planned and implemented the activity and at the same time recorded the implementation of the activity as a video file. Then, they shared their video recordings.

3.2. Research Questions

The study considered teacher candidates' implementation of two different unplugged activities: Storigami (implementation of origami activities through stories) and Coding through Stories. In this respect, this study focused on the following two research questions:

- RQ1: Which Storigami activities were offered to be used in early childhood classrooms by teacher candidates?
- RQ2: Which Coding through story activities were offered to be used in early childhood classrooms by teacher candidates?

In this respect, the study also considered the following research questions (i.e. Table-1) in order to deeply analyze Storigami and Coding through activities.

Table 1. Research Questions

Research Questions
What activities were elaborated by teacher candidates?
What goals were included in the proposed activities?
What skills will be developed through proposed activities?
What materials are needed for the proposed activities?

3.3. Research Design

The study employed a qualitative research design. The activities created by teacher candidates were investigated according to their overall structure and the computational thinking concepts covered.

3.4. Participants

The participants are 15 teacher candidates studying in the Department of Early Childhood Education at a state university in Turkey.

3.5. Data Collection and Analysis

The video recordings shared by teacher candidates were collected and qualitative analysis was conducted to investigate activities. The activities were evaluated based on research questions of the study. That is, each activity was examined with respect to their goals, materials, sub activities as well as the computational skills that activity addressed.

4. RESULTS

4.1. Storigami Activity Sample

Storigami is a combination of origami (i.e., the art of folding paper) and storytelling. As the printed material, the activity only needed colorful papers. In the Storigami activity, teacher candidates tell the story to the audience and at the same time make origami with the story. Each fold or step in the origami process is directly related to an event in the story. While teacher candidates are telling the story, they recorded the storigami as the video file. One sample storigami activity is provided in the following figures.

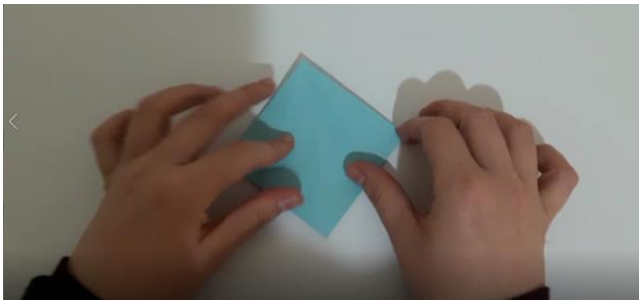


Figure 1. Storigami activity

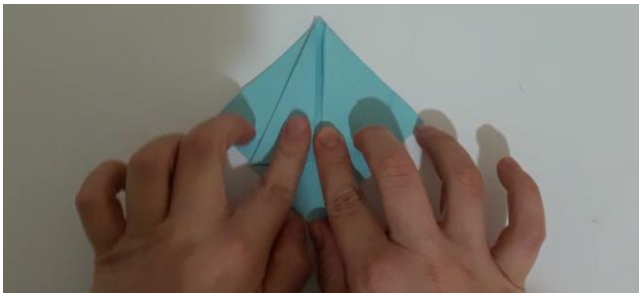


Figure 2. Storigami activity

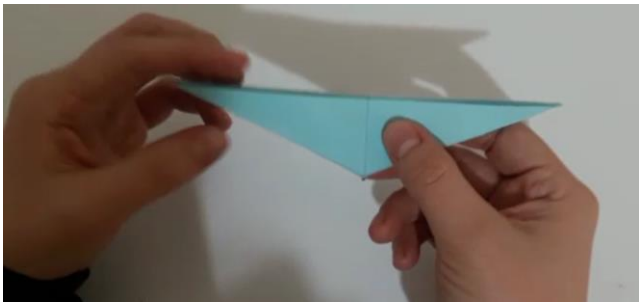


Figure 3. Storigami activity

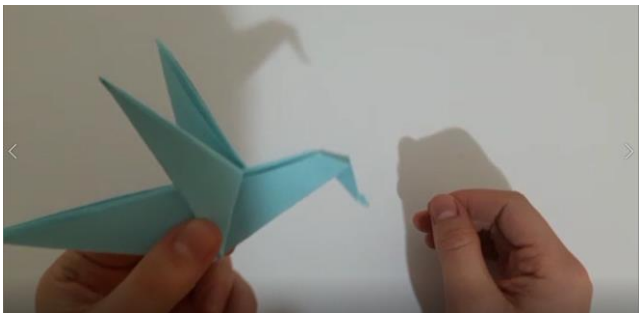


Figure 4. Storigami activity

The story of the activity was as follows:
"In one of the countries there was a blue sky. This sky was so lonely that there was no one around and he was bored. He said that if there is no one, then I will fold and shrink. The sky was very small now, but there was still no one around. Well, he said then, let's do some sports. He folded his right arm, folded his left arm, overturned, folded his right leg, folded his left leg, then suddenly spread his legs, stretched, stretched, and stretched. Turned over, arms outstretched, stretched, stretched, stretched. The sky was still very bored. He said "I wish I was a ship, I wish I was floating in the waters. Maybe I would be a sailboat." At that moment, a bird began to wander in the sky with its huge wings. The sky was very happy when he saw the bird, and now all his troubles are gone."

This activity addressed pattern recognition and algorithm design aspects of computational thinking. Pattern recognition aims to identify similarities or patterns in problems (Barrón-Estrada et al., 2022). In the context of the storygami activity, shapes have some patterns that need to be recognized by listeners. Algorithmic design is for developing steps or rules in order to solve any problem (Wu & Su, 2021). According to the storigami activity, teacher candidates follow and explain an ordered set of steps to create the shape.

4.2. Coding through Stories Activity Sample

In this activity, teacher candidates need papers, colorful tapes, and tree-like materials that can be collected from a forest. In this activity, the purpose is to define a route between bees and a hive. The route includes a step-by-step algorithmic logic and is defined by using the start, forward, turn right, and turn left commands. At the same time, solutions should consider the obstacles in the route. The materials are provided in the printed format as in the following figure.

Figure 5. The materials of the activity

The materials were introduced and the story was provided to kids before starting the activity:
"*It was a warm spring day. Everywhere was full of flowers. Over the flower field, the honeybee was flying. The bee first flew up and down to collect its pollen. She visited the flowers one by one. He then flew two up, one right, and reached the tree. Other siblings came as well. They reached the hive by flying one up and three to the right to make honey for the pollen they collected together. They made such delicious and fragrant honey that the bear, the most honey-loving animal in the forest, smelled the honey. The bear was very hungry. He set out for the hive. When he came to the hive, he asked the bees for a piece of fragrant honey. He began to enjoy the honey given by the bees. This was the most delicious honey the bear had ever eaten.*"

Figure 5. Coding through stories activity

According to this story, the following steps were followed:

1. On the coding material with grids, it was emphasized to the students that every frame is a step, that the steps should be done in sequence (algorithm logic), and that the wrong step would be returned to the beginning.
2. Code blocks (symbols) were introduced to the students.
3. It was emphasized that obstacles are trees and flowers.

Students comprehended "coding", which is the logic of programming, by enjoying and playing games. It was observed that the students enjoyed playing with the material very much.

5. DISCUSSIONS AND CONCLUSION

During the Storigami activities, it was determined that 15 pre-service teachers both improved themselves in addressing students, providing control, being a model, and supporting their students, as well as directing their students' development in "coding". Parallely, in Unan, Aksan & Celikler's study, where the aim was to assess preschool teacher candidates' view on the forming and using living being models through origami for teaching in preschool, it was found out that these activities help facilitate their teaching skills and the students' learning.

In the study, it was determined that pre-service teachers felt good in teaching the subject of coding in the activities they prepared through stories and colorful costumes and materials, and they were able to scaffold the students' learning. It was also concluded that the students learned "coding" easily by hands-on activities using stories, and they participated in the activities with pleasure. Similarly, in a study aiming to introduce algorithm education activities developed on the basis of computer-free coding education for preschool children and to examine the application process of these activities, it was identified that the children participated in the activities fondly, actively participated in the coding activities, and by developing more than one solution proposal to the problem situations, they learned coding and algorithm concepts in an 8-week period (Kucukkara & Aksut, 2021).

5. DISCUSSIONS AND CONCLUSION

During the Storigami activities, it was determined that 15 pre-service teachers both improved themselves in addressing students, providing control, being a model, and supporting their students, as well as directing their students' development in "coding". Parallely, in Unan, Aksan & Celikler's study, where the aim was to assess preschool teacher candidates' view on the forming and using living being models through origami for teaching in preschool, it was found out that these activities help facilitate their teaching skills and the students' learning.

In the study, it was determined that pre-service teachers felt good in teaching the subject of coding in the activities they prepared through stories and colorful costumes and materials, and they were able to scaffold the students' learning. It was also concluded that the students learned "coding" easily by hands-on activities using stories, and they participated in the activities with pleasure. Similarly, in a study aiming to introduce algorithm education activities developed on the basis of computer-free coding education for preschool children and to examine the application process of these activities, it was identified that the children participated in the activities fondly, actively participated in the coding activities, and by developing more than one solution proposal to the problem situations, they learned coding and algorithm concepts in an 8-week period (Kucukkara & Aksut, 2021).

6. ACKNOWLEGEMENT

*This article was produced from the Erasmus + project numbered 2019-1-LT01-KA203-060767 and titled "Future Teachers Education: Computational Thinking and STEAM" of Higher Education Area Strategic Partnership Projects.

7. REFERENCES

Battal, A., Afacan Adanır, G., & Gülbahar, Y. (2021). Computer Science Unplugged: A systematic literature review. *Journal of Educational Technology Systems*, *50*(1), 24-47. https://doi.org/10.1177/00472395211018801

Barrón-Estrada, M. L., Zatarain-Cabada, R., Romero-Polo, J. A., & Monroy, J. N. (2021). Patrony: A mobile application for pattern recognition learning. *Education and Information Technologies*, 1-24. https://doi.org/10.1007/s10639-021-10636-7

Bell, T., Vahrenhold, J. (2018). CS unplugged—how is it used, and does it work? In H. J. Böckenhauer, D. Komm & W. Unger (Eds.), *Adventures between lower bounds and*

higher altitudes (pp. 497–521). Springer. https://doi.org/10.1007/978-3-319-98355-4_29

Bers, M. U. (2008). Blocks, robots and computers: Learning about technology in early childhood. Teacher's College Press, NY.

Deniz, A. & Gönen, M. S. (2020). Developing a scale for evaluation of picture story books: Validity and reliability study. *Journal of Early Childhood Studies, 4*(2), 88-116.

Judge, S., Puckett K. & Çabuk B. (2004). Digital equity: New findings from the early childhood longitudinal study. *Journal of Research on Technology in Education, 36*(4), 383-396. https://doi.org/10.1080/15391523.2004.10782421

Kucukkara, M. F. & Aksut, S. (2021). An example of unplugged coding education in preschool period: Activity-based algorithm for problem solvıng skills. *Journal of Inquiry Based Activities, 11*(2), 81-91.

Işıtan, S. & Turan, F. (2014). Telling story as a narrative analysis approach in assessing children's language development. *Educational Sciences and Practice, 13*(25), 105-124.

Lee, J. (2020). Coding in early childhood. *Contemporary Issues in Early Childhood, 21*(3), 266-269. https://doi.org/10.1177/1463949119846541

Lee, J., & Junoh, J. (2019). Implementing unplugged coding activities in early childhood classrooms. *Early Childhood Education Journal, 47*(6), 709-716. https://doi.org/10.1007/s10643-019-00967-z

McLennan, D. P. (2017). Creating coding stories and games. *Teaching Young Children, 10*(3). https://www.naeyc.org/resources/pubs/tyc/feb2017/creating-coding-stories-and-games.

Nishida, T., Kanemune, S., Idosaka, Y., Namiki, M., Bell, T., Kuno, Y. (2009). A CS unplugged design pattern. *ACM SIGCSE Bulletin, 41*(1), 231–235. https://doi.org/10.1145/1539024.1508951

Resnick, M. (2003). Playful learning and creative societies. *Education Update, 8*(6). http://web.media.mit.edu/wmres/papers/education-update.pdf.

Saxena, A., Lo, C. K., Hew, K. F., & Wong, G. K. W. (2020). Designing unplugged and plugged activities to cultivate computational thinking: An exploratory study in early childhood education. *The Asia-Pacific Education Researcher, 29*(1), 55-66. https://doi.org/10.1007/s40299-019-00478-w

Selby, C., & Woollard, J. (2013). Computational thinking: the developing definition. Retrieved January, 12, 2022 from https://core.ac.uk/download/pdf/17189251.pdf

Tanju Aslışen, E. H. (2021). Storigami in Early Childhood Education: Origami and Story Fellowship. Ankara: Eğiten Kitap.

Ünan, Z., Aksan, Z. & Çelikler, D. (2016). The modelling of living beings with origamı by preschool teacher candidates. *Journal of Research in Education and Teaching, 5*, 165-174.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wyeth, P. (2008) How young children learn to program with sensor, action, and logic blocks. *Journal of the Learning Sciences, 17*(4), 517-550.

Wu, S. Y., & Su, Y. S. (2021). Visual programming environments and computational thinking performance of fifth-and sixth-grade students. *Journal of Educational Computing Research, 59*(6), 1075-1092. https://doi.org/10.1177/0735633120988807

DOI: 10.34641/ctestem.2022.455

Bebras in the Digital Game <Captain Bebras> for Students' Computational Thinking Abilities

Yan-Ming CHEN*, Ju-Ling SHIH
Graduate Institute of Network Learning Technology, National Central University, Taiwan
*peter880118@gmail.com, juling@cl.ncu.edu.tw

ABSTRACT

Bebras is widely known and used to enhance and examine students' computational thinking abilities. In order to make the testing process more intriguing, this study developed a digital game <Captain Bebras> with historical narrative background. This study aims to examine elementary school students' computational thinking abilities playing the game. The digital game simulates the historical events of the Great Voyage time with a map showing various tasks that the player has to perform with computational thinking abilities. Eight themes were classified by Bebras International Computational Thinking, including abstraction, logics, data analysis, decomposition, algorithms, simulation, system evaluation, and generalization. The core theory of each theme is integrated into the game stages, and the content of the Bebras Challenge is also used as the source of the tasks and the scoring base. By comparing students' gaming results with the traditional Bebras Challenge tests before and after the digital game, the researchers investigate students' improvement of their computational thinking abilities and the usefulness of <Captain Bebras>.

KEYWORDS

Computational Thinking, Bebras, Game-Based Learning, Digital Game

1. INTRODUCTION

Computational Thinking (CT) is an indispensable quality in the 21st century (Wing, 2011). Due to the popularization of computers and information, the speed of people's information exchange and problem solving has been greatly improved. It becomes essential for people to keep up with the pace of technological evolution. Therefore, Wing (2006) proposed that CT is a basic skill for everyone and many countries are integrating CT in education.

The United Kingdom added computational thinking to its national syllabus in 2012 (GOV.UK., 2014); Singapore called computational thinking a "national competency"; and the Computer Science Teacher Association (CSTA) proposed a K-12 computer science curriculum in 2016 Computational thinking is listed as one of the five core concepts in the standard. And then US President Barack Obama founded the "Computer Science for All" program (Obama, 2017) to give American primary and secondary school students more knowledge in computer science.

(https://compthinking.csie.ntnu.edu.tw/)

Combining the syllabus plans of various countries in the world, it can be seen that computational thinking is also increasingly valued and important. The "Information Technology" in our syllabus is also based on computational thinking, hoping to cultivate students' logical thinking, systematic thinking and other computational thinking, and to improve the application of computational thinking and problem-solving skills through implementation. In addition to information technology learning in the formal system, the connection with international computational thinking-related activities can also be an important learning inspiration. Therefore, Taiwan has also implemented the "Computational Thinking Promotion Program" for many years, and actively carried out through various channels. Teachers There are empowerment studies, and related activities such as challenges and camps held with Bebras International Computational Thinking for students.

(https://compthinking.csie.ntnu.edu.tw/)

In this study, data will be collected to analyze for the following research questions:

1. Does the game <Captain Bebras> improve learners' conceptual understanding of Bebras Computational Thinking questions?
2. Does the game <Captain Bebras> improve learners' computational thinking performance?
3. What is the learner's overall satisfaction with the game <Captain Bebras>?

2. LITERATUR REVIEW

2.1. Computational Thinking

Computational Thinking means that students use specific strategies to solve problems or understand situations (Selby et al., 2014). It can be seen that this way of thinking is most focused on logic and systematicness, so CT is often used in science, or the integration of knowledge and information with logical judgment, such as the Lightbot game of (Peel & Friedrichsen, 2018) and Scratch programming learning activities, allowing students to visualize the modeling process of mRNA in biological science, and emphasize the five CT concepts of algorithm, abstraction, iteration, branching, and variable in learning.

Computational thinking is not only an ability related to science or information courses, but also other subjects as a way of thinking. It has characteristics such as using computers, data, or modeling to identify, analyze, and implement solutions. It is to help students clarify the logic of problems in a more effective way (Bocconi et al., 2016).

Other than in specific disciplines, computational thinking can be used in every place where there are problems to be solved. Kuo et al. (2020) designed a board game that simulates city life, and added time and money to the game. In the end, it was found that even in the absence of electronic products, or the absence of discipline-related themes in the

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

board game, students' CT abilities were improved with the board game.

Therefore, computational thinking is a problem-solving process. It uses logical processes such as decomposition, thinking, logic, and algorithm to produce solutions. When we encounter problems in life, we will try to disassemble the problem, find elements of small problems, abstract them into rules or principles, and then list various algorithms that can help to achieve goals. The purpose of writing programming languages is to perform computational thinking to solve problems.

2.2. Bebras

Bebras was conceptualized by Valentina Dagiene of Vilnius University. Bebras means "beaver" in Lithuanian, and refers to the prospect of having the students to study hard like beavers to achieve their goals with the intentions of diligence, intelligence, liveliness, and challenges.

The Bebras Challenge started on September 25, 2004, and is held globally in the International Bebras Week in mid-November every year. Since then, more European countries have joined, and Taiwan has officially participated in 2012. The Bebras Challenge hopes that the students participating in the competition can apply computational thinking to solve problems in their lives, so most of the questions are designed to be situational questions, and the operational thinking is divided into eight themes corresponding to the questions of the challenge (https://www.bebras.org/).(Figure 1)

1. Abstraction: Identify and extract relevant information to define the subject, simplifying things by removing unnecessary details.
2. Logic: Prediction and analysis to help understand things to clarify facts.
3. Analyze data: observe and understand through the collected data.
4. Decomposition: Decompose the problem into smaller and understandable problem content.
5. Algorithm: Create a logical flow of steps to use to solve a problem.
6. Simulation: Build an environment model similar to the real world.
7. Systematic evaluation: make judgments objectively and systematically.
8. Generalization: After observations, try to build rules to predict outcomes.

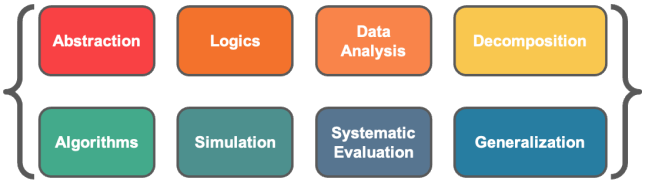


Figure 1. Eight Bebras CT aspects

From the statistics of the number of applicants for the Bebras Challenge, it can be seen that applicants in each group increase year after year. The latest year of 2021, it broke the record of 217,000. Among them, the Cadet group, referring to the seventh and eighth grades of middle school, is the largest group among all. Thus, the popularity of computational thinking education has been growing. Three goals of the Bebras Challenge are defined as follows. (https://bebras.csie.ntnu.edu.tw/)

1. Stimulate students' interest in learning information science. Bebras Challenge not only helps teachers to understand students' computational thinking ability, but also hopes to introduce the basic concepts of information science to students through situational tasks to stimulate their interest in learning. It is to let students understand that the application of information science concepts can be seen everywhere in life. The problem-solving and reasoning method can also improve students' motivation and enhance their ability of high-level thinking.
2. Improve students' ability to use computational thinking to solve problems. Bebras Challenge uses life situations such as family life, group cooperation, work arrangements, etc. to guide students to think and solve problems. The questions focus on computational thinking and problem-solving skills that only basic knowledge is required. Through the questions, students can understand that many problems in life can be solved by computational thinking.
3. Reduce students' fear of information science. Bebras Challenge concretizes abstract information science knowledge and presents it in situations that will be encountered in daily life. Thus, students who have not had information science education can also use their logic, induction, reasoning, and operations skills. On the other hand, the content of the questions is interesting which helps reduce students' anxiety about information science learning.

2.3. Digital Game-Based Learning

Technology products have indirectly affected educational models due to its convenience, multimedia effects, and fun. Digital game-based learning has brought learners strong motivations to learn and enhance students' attention more than ever before (Becker, 2007; Pivec, 2007).

By integrating knowledge from the textbooks into the game, tasks can provide students a simulated scenario with situations and stories that enable students' acceptance of knowledge and learning motivation (Chen & Lin, 2016).

Innovative game-based learning includes physical board games, STEM or robotics and other technology-integrated games, such as an interdisciplinary board game designed by Shih (2017) using the historical context of the Age of Discovery, also extended to integrate robots to enhance students' CT skills .

Game-based learning is not necessarily limited to the application of physical games or disciplines, but also strategies, question-answering, and feedback mechanisms are added to the game (Rojas-Mancilla et al., 2019). Therefore, digital games have been applied to many studies. For example, the game system TAPASPlay (Turchi et al., 2019) combines digital games with computational thinking. It is found that learners can increase their interests in computational thinking through games. Sobrino et al. (2020) designed a game system called Robotic, which uses real-life roads and traffic signs to simulate the real-world environment in the game, and promotes the cultivation of primary school students through various tasks in the game. Interest in computational thinking, and learning program logic, etc.

However, learning is not a cure-all, and learners may not necessarily feel the improvement of their own abilities or the application of knowledge after they play a game. For example, Shih (2016) used Robotic Game to analyze the code path and found that after multiple rounds of play, the learners' performance of game mechanics and operational thinking tended to be stable. Only players can really use their own experience to think, or effectively apply resources and strategies to the game.

3. GAME DESIGN

3.1. Game Mechanism

The design background of "Captain Bebras" comes from the 16th century era of great voyages. Magellan set sail from Spain in search of the spice road. Its story setting is inspired by our research team's past interdisciplinary history board games (Huang et al, 2019) , At that time, a large-scale world map (600x400cm) was used to fully present the territorial scope of the great voyage era in history, and students controlled robots through programming to gradually complete tasks with computational thinking.

In this study, "Captain Bebras" is a digital learning game that integrates Bebras' computational thinking. By referring to historical facts, Bebras' international standard of computational thinking and innovation of game elements, a series of tasks that simulate the real world are designed. The results were brought back to Spain as tribute. In the game, players take on the role of Captain Beaver on a voyage. In the process, they will encounter five different simulation problems, and gradually learn the corresponding topics of computational thinking through the game challenge process from simple to difficult.

The five missions in the game are designed according to the topic classification of Bebras with two missions in the simple level, two missions in the medium level, and one mission in the difficult level.

First, the player starts with Mission 1 ─ Direction Decoding (Figure 2). The player role-playsCaptain Bebras, and moves to left, right, and forward according to the programming codes trying to find the correct treasure chest. This mission uses Algorithms and Systematic Evaluation skills in computational thinking.

After the player gets the treasure chest from the correct tree in the first mission, the player enters Mission 2—Spices Maze (Figure 3). In this mission, the player summarizes the order of spices in the limited scroll clue, and tries to find the correct order with his/her reasoning ability. Then, s/he can choose the correct maze map. The computational thinking skills used in this mission includes Logic, Algorithms, Simulation, Abstraction, Generalization and Systematic Evaluation.

After successfully obtaining the correct maze map, the player continues to Mission 3 - Spices Purchase (Figure 4). S/he needs to convert the complex maze map into intuitive numbers, and follows the sequence to various places for spices purchase. The computational thinking skills used in this mission includes Abstraction, Decomposition and Systematic Evaluation. When the player successfully completes the above three tasks, a staged transition screen will be given to bring s/he to another scenario.

After completing the first three missions, the player goes to a new story map to challenge more difficult levels, Mission 4 –Preference Combination (Figure 5). In this mission, the player sees a spice preference chart and chooses from it. Every spice can only be purchased in one country so s/hehas to find the most favorable purchase combination before taking action onto the map of Mission 4 (Figure 6). This mission uses computational thinking skills including Algorithms and Systematic Evaluation.

In the end of the story, the player returns to Spain for Mission 5 - Tribute (Figure 7). and the player needs to use the spices purchased earlier to pay tribute to each castle in Spain by using the thinking of Euler Circuit. S/he would finally return to the start point and complete all the missions of <Captain Bebras>.

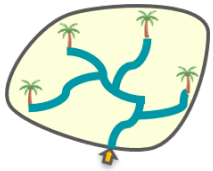


Figure 2. <Mission 1>

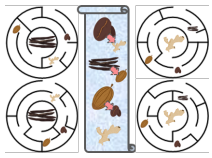


Figure 3. <Mission 2>

Figure 4. <Mission 3>

Figure 5. <Mission 4-1>

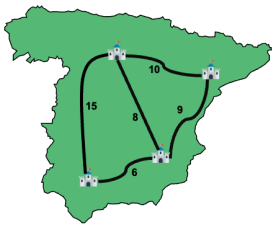


Figure 6. <Mission 4-2>

Figure 7. <Mission 5>

Each task in the game mission of this study is designed with reference to the Bebras computational thinking aspects, so the players' abilities can be analyzed accordingly. For example:

1. Abstraction: In Mission 2 and 3, the player extracts relevant information from the prompts and deletes unnecessary information to simplify the goal. For example, s/he has to find the correct maze map by extracting the information of the clues. The goal of mission three can only be completed by extracting information from the complex maze before transfering onto the map.
2. Logics: In Mission 2, the player uses reasoning to analyze the spice and order information before choosing the map.
3. Data Analysis: In Mission 5, the player observes all the path lengths in order to find the shortest path by using Euler circuit.
4. Decomposition: In Mission 3, the player breaks down the purchase problem into defining the order of spices, searching for spices on the map, and defining the path sequence in order.
5. Algorithms: In Mission 2, the player uses algorithmic commands to find which tree the treasure chest is in, then defines the spice codes for each path, and finally finds the matching path. In Mission 4, the player needs to find the best combination from the chart before acting on the map using the combination.
6. Simulation: In Mission 2, the player needs to use the limited amount of gold coins s/he has for the task just as in real life. In Mission 5, the player needs to consider the length of each path before finding the shortest path as in the real life choices.
7. Systematic Evaluation: In all Missions, the player needs to make judgments objectively with logical thinking in order to achieve the task goals.
8. Generalization: In Mission 2, the player observes the rules on the clues, establishes the order of spices, and generalizes the sequence for the application of the maze.

3.2. Gaming Process

Before the game, the teacher/game master (GM) starts a 10-minute game introduction, followed by a 15-minute pre-test. Then let the students start to play the game <Captain Bebras>. There are 5 tasks in each round of the game. It is estimated that one round of the game will take 60 minutes.

After completing the first round, a mid-term test will be conducted for 15 minutes. After entering the second round of the game, the teacher/ GM provides feedback to the students, about 10 minutes, and a post-test for 15 minutes, so the total game time is about three hours.

4. RESEARCH DESIGN

4.1. Research Process

<Captain Bebras> is designed for the Cadet group so this study aims to find 20 junior high school students who are between the ages of 13 and 15. A computer classroom will be used for the game-based learning class, and each student will use one computer individually. Students will be assigned an individual account for the game. The research process is shown in Figure 8.

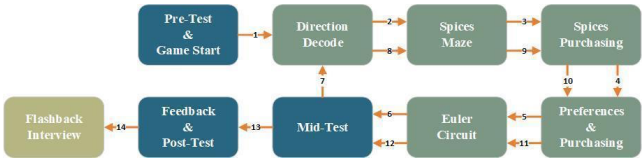


Figure 8. Research process for game-based learning with <Captain Bebras>

4.2. Research Data Collection and Analysis

In this game, students will need to:

1. Understand the game scene and game mechanism.
2. Complete game missions and express mission objectives.
3. Challenge the difficult tasks of the game.
4. Find out a strategy or solution to accomplish the task.
5. Infer what is learned from the game to life situations.

Wing (2006, p.33-35) suggested that "Computational Thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent." Therefore, when students play the game, the teacher should pay more attention to the students' problem-solving process instead of just the performances. In the research analysis stage, both students are invited to reflect on the gaming process, and understand in-depth the students' CT skills performances and problem-solving processes.

5. EXPECTED RESULTS

5.1. Bebras Performances

The game design ideas for all the levels in the game <Captain Bebras> are derived from the Bebras official exam questions. The design and research of gamification are carried out for the Cadet groups, referring to the seventh and eighth grades of middle school, classified by the official Bebras.

The first research question explored in this study is "Does this game <Captain Bebras> improve learners' conceptual understanding of the Bebras Computational Thinking

questions?" Therefore, Pre-test, conducted before playing <Captain Bebras>, mid-test following Level 3, and post-test while finishing all levels of <Captain Bebras> game. It is hoped that through the test at different times, learners' conceptual understanding to the Bebras computational thinking can be seen due to playing the game <Captain Bebras>.

The level of all the questions in the three tests are the same, but the question contents are not repeated. The calculation of the score and the test time are different from the official method, as shown in Table 2.

Table 2. Test method comparison

	Official Scoring Method	Scoring Method For This Study
Test Leveling	Simple, Medium, Hard	Simple, Medium, Hard
Total Questions	15	5
Examination Time	45 mins	15 mins
Lowest Score	60	0
Highest Score	300	100
Deduct Points	Yes	No

In the tests, the students' completion and scores will be emphasized (Figure 9). If the test completion rate of the pre-test is 40% and the post-test is 80%, it can be inferred that this game has missions that promote students to understand Bebras computational thinking better. Furthermore, if the student's pre-test score is 60 points and the post-test score is 100 points, it can be speculated that this game can promote students to understand the eight kinds of computational thinking of Bebras, and successfully use them in solving problems. Therefore, from the pre-, mid-, and post-test, if the learners have significant improvements in the degree of completion and accuracy, it shows that the students have a clearer understanding of the concept of Bebras computational thinking due to the game (Figure 10).

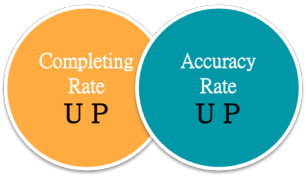


Figure 9. Test Aspects

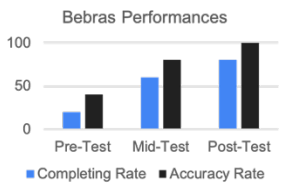


Figure 10.Test Scores

5.2. *CT Performances*

The second research question discussed in this study is "Does this game <Captain Bebras> improve learners' computational thinking performance?" The data source for this question is the same as that of the first research question using the three tests. Each official Bebras test question has its corresponding standards for the classification of grades and the operational thinking ability of the test questions.

The operational thinking performance discussed in the second research question refers to Bebras' eight operational thinking abilities: Abstraction, Logics, Data Analysis, Decomposition, Algorithm, Simulation, Systematic Evaluation, and Generalization. Since each question will correspond to the multi-faceted CT abilities, we may observe multiple questions whether students have improved or changed the operational thinking ability in eight different aspects during the process of pre-, mid- and post-test.

This study expects that by playing the game <Captain Bebras>, learners can gradually improve their computational thinking skills so their completion rate and speed would accelerate.

5.3. *Overall Satisfaction*

This study expects that most of the learners like the game <Captain Bebras>, and are more interested and motivated in computational thinking because of the game-based learning. Through a series of missions that simulate real life problems in this research, learners can better understand the concept of computational thinking, and then apply computational thinking in real life situations.

With high improvement and satisfaction levels, it shows the potential of CT games.

6. CONCLUSION

In the 21st century, the advancement of information technology has led to changes in teaching methods. Schools no longer teach by rote learning, but use modern technology and novel learning models to help students build computational thinking skills. More and more attention is paid to students applying the knowledge they have learned to real life, so that students can learn all the time, both inside and outside of school.

Today's society places great importance on computational thinking. Computational thinking is a kind of thinking mode, especially for problem-solving. It is to apply logical thinking and finding solutions to complex problems through disassembly, methodical sorting, analysis, deduction, and other thinking methods in problem solving. Computational thinking can be used not only in disciplines, but also in daily life. Its essence is to decompose a big problem that is difficult to solve at once, but to dissect into small tasks to seek combinational solutions..

In this study, <Captain Bebras> guides learners to a digital game that simulates real world problems. Because Bebras computational thinking is incorporated into the game, learners are in the process of computational thinking when they try to decipher the tasks. In the end, after the game is completed, learners can apply them in real life more proficiently than in the past.

This study expects that the game <Captain Bebras> can increase learners' curiosity and motivation for computational thinking. Because of the incentives and novel learning modes of the game itself, it can be expected to include more

Bebras challenge questions into games to help children learn computational thinking. Digital games can also be designed to meet understanding levels for different age groups so as to tap the potential of each child's computational thinking.

7. REFERENCES

Becker, K. (2007). Digital game-based learning once removed: Teaching teachers. *British Journal of Educational Technology, 38*(3), 478-488. https://doi.org/10.1111/j.1467-8535.2007.00711.x

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing computational thinking in compulsory education. European Commission, *JRC Science for Policy Report, 68*,15-17.

Chen, H. R., & Lin, Y. S. (2016). An examination of digital game-based situated learning applied to Chinese language poetry education. *Technology, Pedagogy and Education, 25*(2), 171-186.

GOV.UK. (2014). *National curriculum in England: computing programmes of study*. Retrieved July, 16, 2014.

Huang, H. Y., Huang, S. H., Shih, J. L., Tsai, M. J., & Liang, J. C. (2019). Exploring the role of algorithms in elementary school students' computational thinking skills from a robotic game. *In 3rd International Conference on Computational Thinking Education, CTE 2019* (pp. 217-222). The Education University of Hong Kong.

Kuo, W. C., & Hsu, T. C. (2020). Learning computational thinking without a computer: How computational participation happens in a computational thinking board game. *The Asia-Pacific Education Researcher, 29*(1), 67-83.

Obama, B. (2017). *Computer science for all.* Retrieved from WhiteHouse: https://www.whitehouse.gov/blog/2016/01/30/computer-science-all.

Peel, A., & Friedrichsen, P. (2018). Algorithms, abstractions, and iterations: teaching computational thinking using protein synthesis translation. The American Biology Teacher, 80(1), 21-28.

Pivec, M. (2007). Play and learn: potentials of game-based learning. *British journal of educational technology, 38*(3), 387-393.

Rojas-Mancilla, E., Conei, D., Bernal, Y. A., Astudillo, D., & Contreras, Y. (2019). Learning Histology Through Game-Based Learning Supported by Mobile Technology. *International Journal of Morphology, 37*(3),904-906.

Shih, J.-L. (2016). Computational Thinking in the Interdisciplinary Robotic Game: the CHARM of STEAM. In S.-C. Kong & Harold Abelson (Eds.). Computational Thinking Education in K-12. Boston: MIT Press.

Shih, J. -L., Huang, S. H., Lin, C. H., & Tseng, C. C. (2017). STEAMing the Ships for the Great Voyage: Design and Evaluation of a Technology integrated Maker Game. IxD&A, 34, 61-87.

Schez-Sobrino, S., Vallejo, D., Glez-Morcillo, C., Redondo, M. A., & Castro-Schez, J. J. (2020). RoboTIC: A serious game based on augmented reality for learning programming. *Multimedia Tools and Applications, 79*(45-46), 34079-34099.

Selby, C., Dorling, M., & Woollard, J. (2014). *Evidence of assessing computational thinking*.

Turchi, T., Fogli, D., & Malizia, A. (2019). Fostering computational thinking through collaborative game-based learning. *Multimedia Tools and Applications, 78*(10), 13649-13673. https://doi.org/10.1007/s11042-019-7229-9

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2011). Research notebook: Computational thinking—What and why. *The Link Magazine*, 20-23.

DOI: 10.34641/ctestem.2022.456

A Robotic-based Approach for CT Development: Challenges of Teaching Programming Concepts to Children and the Potential of Informal Learning

Rafael ZEREGA[1*], Ali HAMIDI[2*], Sepideh TAVAJOH[3*], Marcelo MILRAD[4*]
[1,2,3,4] Faculty of Technology, Linnaeus University, Sweden
rafael.zerega@lnu.se, ali.hamidi@lnu.se, st222yd@student.lnu.se, marcelo.milrad@lnu.se

ABSTRACT

In many countries worldwide, Computational thinking (CT) is now considered as a fundamental skill for dealing with the challenges of the 21st century society. One of the most common ways of imparting CT knowledge in K-12 education is by teaching programming and coding, as it requires applying a set of concepts and practices that are essential for thinking computationally. However, learning to program can be challenging and it may take time to develop these skills in the context of school activities. Thus, complementing formal K-12 education with after-school or other types of informal learning activities aimed at fostering CT concepts and practices among young students can be an alternative approach to develop these skills. During the summer of 2021, we carried out a series of workshops in the context of a summer camp taking place at a public library, organized by a local municipality in southern Sweden. These workshops (with a total teaching duration of 20 hours in one week) consisted of activities where children aged 11-14 had to assemble wheeled robots and then program them using a visual language to make them execute different types of tasks and challenges. The outcomes of our study show that roughly one third of the participants managed to program the robots with code that made use of CT core concepts, such as conditionals, loops, and logical operators, among others. The rest of the children did not manage to successfully apply these concepts and thus they could only manage to program sequential linear scripts. We argue that learning to program and understanding some of the main CT concepts, which are for the most part very abstract, is a process that takes time and thus, extracurricular activities can be an effective method to complement formal education and help young students develop their CT and programming skills.

KEYWORDS

Computational Thinking, Computational Concepts, Computational Practices, Programming, Informal learning.

1. INTRODUCTION

Computational Thinking (CT) is a thought process focused on problem-solving that is deemed by many researchers and policymakers as a fundamental skill for dealing with the challenges of the 21st century society. Wing (2006) brought CT to public attention explaining the essence of this concept and advocating for its inclusion in the K-12 curricula. Grover & Pea (2018) argue that CT comprises a set of concepts and practices that are required for formulating a problem and expressing its solution effectively. Lu & Fletcher (2009) are more emphatic about the relevance of teaching CT in schools and argue that it should be taught to every student along with the other *three R's* (reading, writing and arithmetic). As a result of all this advocacy, an increasing number of countries around the world have started to impart knowledge related to digital competence, CT, and programming, as part of their K-12 curricula. Although some authors argue that CT is not solely about programming (Grover & Pea, 2018; Wing, 2006), learning algorithm design to program a computer, a robot or any other computerized machine is an essential skill when attempting to solve a problem through a computational solution, which is one of the main goals of CT (Grover & Pea, 2013).

However, to grasp the true nature of CT and to be able to program a computational artifact, there is a set of concepts and practices that must be understood (Brennan & Resnick, 2012; Grover & Pea, 2018). Some of these concepts are relatively abstract; concepts like algorithmic thinking and sequences, using conditionals, applying loops to repeat a given set of actions and storing data in variables, to mention just a few, could at first be somewhat difficult to fully grasp. Regarding CT main practices such as problem decomposition, iterative refinement, as well as testing and debugging, among others, the situation is not much different. For instance, some studies suggest that novice programmers in K-12 education face difficulties when attempting to detect and debug errors in their code (Carter, 2015; Haduong & Brennan, 2018). Similarly, other studies have focused on some of the common misconceptions regarding programming concepts and the difficulties that young students encounter when starting to learn how to program (Grover & Basu, 2017). Yet another study from Sanders & McCartney (2016) identified a few thresholds programming concepts that tend to be problematic for novice young programmers. Learning the basics of programming can therefore pose several challenges and that is why some authors suggest that this knowledge should be imparted at a very early age. Some scholars advocate for introducing children to CT and programming concepts as early as kindergarten education (Fessakis et al., 2013; Sullivan & Bers, 2016). Furthermore, Lu and Fletcher (2009) argue that students that have been introduced to CT at an early age tend to show higher probability of successfully learning more advanced programming later.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

There is, nevertheless, a limited number of hours in the school curricula that can be dedicated to imparting these new subjects and therefore finding other instances to teach CT and programming to young students could be an effective way to help students learn programming concepts and practices. Informal learning activities addressing CT, such as after-school workshops can be an effective manner to complement K-12 formal education and let children acquire additional knowledge in this subject (Ker et al., 2021). In this paper we argue that informal learning activities can offer students the possibility to further explore and test programming concepts and practices, allowing them to deepen their understanding of these matters in a friendly environment and without the stress normally associated to formal education as extracurricular activities are not subject to evaluation in form of official grades. To test the potential of informal learning to foster and develop CT skills among young students we conducted a series of workshops during one week with a group of young students aged 11-14 that had little or no previous experience in programming. These workshops were conducted at the main public library in a city in southern Sweden. We used educational robots, Engino ERP[1], that the children had to assemble and then program so that they would execute a series of tasks and challenges. Considering all the above, we defined two research questions that guided this study: (1) *What are the main challenges when teaching CT concepts and practices to youngsters with little or no previous programming experience?* (2) *What is the potential of informal learning spaces as an alternative for complementing the teaching of CT and programming concepts provided by formal K-12 education?*

The rest of this paper is organized as follows: in section two we provide a background regarding the relation between CT and programming. Section three provides a description of the methodology used for this study. Section four presents the main results and lastly, section five ends this paper presenting our discussions and conclusions on the results.

2. THE ROLE OF PROGRAMMING IN CT

Many countries around the world are currently in the process of modifying their K-12 educational curricula to develop so called *digital competences* (Heintz et al., 2017) and therefore CT has increasingly gained more attention. CT was originally coined by computer scientist, Seymour Papert in his book *"Mindstorms: Children, computers and powerful ideas"* (1980). Papert was one of the pioneers of constructionism, a constructivist learning theory where students create knowledge by exploring, constructing, and testing. This is the reason why building plays a central role in constructionism. CT derives from this learning theory and consequently one of its main objectives is to design and build systems (Wing, 2006). Cuny et al., (2010) further developed the definition of this concept by explaining that CT is a thought process required to formulate a problem and to express its solution in an effective way so that it can be carried out by an information processing agent (such as a computer or even a person). Being able to instruct or program a computerized system is, therefore, a fundamental skill within CT (Grover & Pea, 2018; Kynigos & Grizioti, 2018).

To fully understand the relevance of programming within CT it is necessary to analyze how programming relates to CT and computer science (CS) as a whole. As can be observed in Figure 1, CT and CS are two fields of study that overlap only partly. In other words, CT is not solely about CS (and vice versa). Programming (coding) lies in the intersection between these two worlds and thus it is an important component of CT (Angevine et al., 2017). Although Wing (2006) argues that CT is an approach to problem-solving that is considerably broader than mere programming, she later clarifies and further develops this concept by explaining that CT is a thought process involved in formulating a problem and its solution so that this solution will be effectively carried out (Wing, 2011). CT means, therefore, using computer-based solutions to solve real-world problems and consequently programming is an essential skill necessary for applying CT.

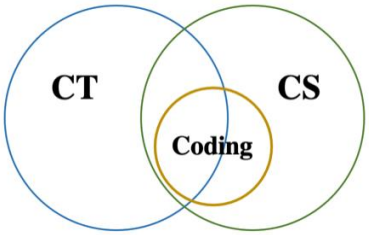


Figure 1. The relationship between Computer Science, Computational Thinking and coding (Angevine et al., 2017)

When considering the importance of programming and algorithm design within CT it is then necessary to consider what different authors call the concepts and practices of CT (Brennan & Resnick, 2012; Grover & Pea, 2018). Other authors use different terms such as CT skills (Mills et al., 2021) and when examining carefully all these terms it is not rare to find some authors using them interchangeably. However, regardless of the exact term used to refer to these different dimensions of CT, there is something that they have in common: they are all directly or indirectly related to the process of programming and building algorithms. Being able to have a good understanding of what these concepts and practices are all about is therefore essential to understand the process of designing algorithms and programming computerized devices. In this study we aim to analyze how the participants of the workshops mentioned earlier managed to make use of these CT concepts and

[1] https://www.engino.com/w/

principles when they were assembling and programming robots.

3. METHODOLOGY

In this section we present core aspects regarding the design of the study in terms of the workshops' settings, the participants, the technological equipment used for teaching and the type of data collected for the later analysis.

3.1. *Settings and Participants*

This study was carried out based on a series of workshops that took place in late June 2021. These workshops were conducted at the main public library in a city located in southern Sweden to kick off the summer vacation. Five workshop sessions were held, from Monday to Friday, each session lasted for four hours. The participants were seven boys and two girls aged 11-14 who, although having received education on digital competence in school, they had little or no previous experience in programming. The workshops were led by two tutors in charge of explaining the topics to be learned during each session and helping the participants in case they request assistance (see Figure 2).

3.2. *Workshops Design and Theoretical Foundation*

As mentioned above, for this study we conducted five workshops, one every day from Monday to Friday. Each workshop had a duration of four hours (a total of 20 hours for the entire workshop series). Each student was given a laptop computer where they could visualize the assembly instructions and run the software required for building the algorithms to program the robots. The activities carried out in each of the five workshop sessions were as follows: (1) assembling the robots, learning about the sensors and creating simple linear algorithms, (2) using loops and conditionals in algorithms and learning about Boolean data type, (3) learning more about conditionals, using logic and arithmetic operators and using Integer data type, (4) using variables and deepening on the use of loops and conditionals, and (5) free practice and testing what has been learned during the workshop series.

As for the programming-related activities, the main objective was that children would make use of the different types of sensors (ultrasonic sensors, infrared sensors and color sensors) and by designing algorithms they would program the robots so that they would interact with their surroundings and execute tasks such as, avoiding obstacles, following the borders of a path, deciding to turn left or right based on the clear space available on each side, among others. To instruct the robots for executing such tasks, the children would have to build algorithms that make use of programming concepts such as conditionals and loops, as well as using logical and arithmetic operators, among others. For this purpose, at the beginning of each workshop and before starting with the actual hands-on activities of the day, the tutors gave a brief keynote presentation where they introduced the students to different concepts of programming, explained how robots interact with the physical world and to which extent they are present in our daily lives.

The assessment of the learning process for each workshop session was based on observing whether the robot was executing the task that the children had intended to program and by analyzing the actual algorithms that they had made using the block-based programming platform provided by the Engino ERP.

These workshops were designed taking in consideration the notions of constructionism, aimed at offering student-centered activities and allowing children to explore and test their ideas through building and collaborating with their peers and instructors (Papert & Herel, 1991). Constructionism, pioneered by Papert, puts the emphasis on allowing students to generate their own knowledge by building and experimenting while the educator plays the role of a consultant or coach. The idea was that during the workshop series the participants could learn about robots by showing them through examples that robots and other types of automated devices are increasingly present in our current society. By giving the children the chance to assemble their own robots and program them, so that they can interact with the environment, the children could not only learn CT concepts, but also get an insight on how robots work as well as understanding what is their potential to improve our lives and what are the risks associated with this technology.

Figure 2. Workshops at the public library

In addition, based on the ideas from Laurillard (2013), we regarded the process of teaching as a design science. For this study, the design of the workshop series was done creating learning activities based on the concepts of the TPACK (technological pedagogical content knowledge) framework for the effective use of technological tools to support and enhance the learning process (Mishra & Koehler, 2006).

3.3. *Educational Equipment*

For this study we used a set of educational programmable robots called Engino ERP, which is targeted to kindergarten, elementary and secondary students (depending on the model). Engino ERP is a line of construction kits that use various sensors that allow the user

to build and program robots that can interact with their surroundings. These robots can be programmed using a special software that offers a block-based programming environment to let children build algorithms in a syntax-free coding mode. The Engino ERP includes a wide range of sensors that allow the robots to execute different types of tasks as they measure different parameters from the environment. For this workshop series the children worked with three types of sensors: infrared sensor, color sensor and ultrasonic sensor.

3.4. *Data Collection and Assessment*

During all five workshops the researchers took field notes, photographs, and screenshots of the computers where the children were building their algorithms to program the robots. This data was analyzed using a qualitative approach to identify which were some of the most challenging computational concepts and principles in the process of learning to build and program the robots. By analyzing the children's code and the performance of their robotic creations we attempted to get an insight regarding how the children managed to use fundamental programming and CT concepts such as algorithmic sequences, conditionals, loops, and logical operators, among others. To assess the learning progress of the children during the workshops, we took into consideration the CT concepts and practices defined by Brennan & Resnick (2012).

4. FINDINGS

This section will present the most relevant findings based on the data collected during the workshop series. We divided these findings into two areas: (1) *physical assembly* and (2) *CT and programming.*

4.1. *Physical Assembly*

Two types of ERP sets were used during the workshop activities in order to explore how constructing methods influence the children in terms of their CT practices, such as being incremental, reusing and remixing, modularizing, testing and evaluating. The children worked with semi-built robots that were to be completed and modified either by following the step-by-step 3D instructions, that they had on their computers, or by resorting to their own inventiveness. All nine participants preferred to build the robots based on their own inventive ideas rather than by following the instructions. The children showed more engagement when constructing their own creations. Several children mentioned that building freely was more amusing than building by following instructions. In addition, the children tended to lose both interest and focus when they faced a situation where the assembly process was particularly difficult. A big challenge that the children faced in terms of the physical assembly was to find the best way to mount and position the sensors on the robots so that they would get an accurate reading of the surroundings. Whereas some children would become frustrated and annoyed when they could not manage to position the sensors correctly to get an accurate reading, others were particularly motivated to test many times until they found the best way to position the sensors.

4.2. *CT and Programming*

As mentioned earlier, the children participating in these workshops had practically no previous experience doing any type of programming. The brief keynote presentation that took place at the beginning of every workshop in combination with the hands-on activities allowed all the children to get a rough understanding of what an algorithm is. All nine children managed to design simple linear algorithms that could instruct the robots to execute simple tasks such as going forward, turning right, left, and stopping. However, only four children managed to successfully design algorithms that made use of some of the main computational concepts, such as conditionals, loops, and logical/arithmetic operators (see Table 1). Without these programming concepts, the robots could only be instructed to execute a fixed sequence of actions (linear algorithm in Table 1), but they would not be able to interact with the environment in any way.

Table 1. Types of Computational Concepts that the participants managed to successfully use in their algorithms (*yes* means used successfully).

Student ID	Linear Algorithm	Loop	Conditional	Logic and arithmetic operators	Variables
1	Yes	No	No	No	No
2	Yes	Yes	Yes	Yes	No
3	Yes	Yes	Yes	Yes	No
4	Yes	No	No	No	No
5	Yes	No	No	No	No
6	Yes	Yes	Yes	Yes	Yes
7	Yes	No	No	No	No
8	Yes	No	No	No	No
9	Yes	Yes	No	No	No

As for the use of sensors, among the children that used sensors in their robots, the one that was used the most was the infrared sensor. The children mentioned that it was fun to use this sensor because it allows them to do many different tasks with it and it was easy to set up. The other two sensors (color sensor and ultrasonic sensor) were used very seldom. According to the children, the color sensor was hard to use because the calibration process to set it up required a considerable amount of trial and error to get it working correctly. The ultrasonic sensor, although easy to set up because it did not require any type of calibration, was used successfully by only one of the participants. It is important to mention that the infrared sensor uses Boolean data type (data that has one of two possible values: true/false). The color and ultrasonic sensor, on the other hand, use Integer data type (in this case positive whole numbers and zero). According to the children, working with Boolean data was easier and more straightforward than working with Integer data, which may explain why only few students managed to successfully use the color and ultrasonic sensor.

In the next and final section, we present our discussions and conclusions based on the data we collected during the workshop series. We divided it into five subsections to make it easier to connect the discussions with the topics of the research questions that guided this study.

5. DISCUSSIONS AND CONCLUSIONS

5.1. *The Importance and Challenges of Learning CT Concepts*

Learning to program requires being able to understand a set of CT concepts and practices, which can be a challenging and long process. Taking into consideration the computational concepts defined by Brennan & Resnick (2012), such as sequences, loops, and conditionals, we can notice that after having completed the workshops, all nine participants understood that an algorithm is an expression of a sequence of individual instructions that a computerized machine executes. Indeed, all nine children managed to instruct the robots to execute tasks such as making it move forward and then turn at certain points to describe, for example, a geometrical shape (linear algorithms). However, only three children managed to successfully design an algorithm that would include a computational concept that would allow the robots to use the data coming from the sensors to be able to interact with the environment. Basic control structures such as *if-else conditionals*, *for loops* and *while loops* are fundamental computational concepts to have a robot or other computerized machine make decisions based on the information that is coming from the sensors. Learning to build algorithms using these programming concepts will not only allow the robots to interact with the environment but it will also serve the children as a means of exploration and a way to create and express computer-based solutions to real-world problems. Engaging in programming offers young students the possibility to exercise a set of different computational concepts and higher order thinking skills, such as reasoning, analyzing and evaluation (Falloon, 2016).

5.2. *The Challenges of Building and Debugging Collaboratively*

The results based on the workshops we conducted suggest that teaching programming concepts poses some difficulties. Not only was it challenging for the students to fully understand some of the programming concepts as they struggled when asked to explain their own algorithms, but it was relatively hard for them to be able to identify the origin of the problem in their code when the robot was not able to execute the task successfully. The difficulties in detecting and debugging errors in the code of novice programmers are not uncommon and they occur even among students at college level (Carter, 2015). The situation is more evident among young students in primary and secondary schools. Haduong & Brennan (2018) argue that best practices of code debugging are, for the most part, undefined in K-12 education. Not being able to fix a piece of code can be extremely frustrating and demoralizing for young students and therefore it is extremely important to understand the relevance of teaching young students some basic rules regarding how to identify and debug errors in the code and other types of problems that may arise. The physical assembly was too a relevant aspect of the CT development and a challenge for the children as this activity required them to build together, communicate and exchange ideas, for example, when the children were discussing the best way to position the sensors, and when they were evaluating their construction methods and results.

5.3. *The Convenience of Block-based Programming for Novice Programmers*

Programming activities that use block-based programming for introducing children to CT and algorithm design are a good choice as the graphical interface allows them to build algorithms and focus on computational concepts and practices without the need to take care of the syntax associated with text-based programming. Mladenovic et al. (2018) sustain that most novice programmers often focus on the syntax of the programming language instead of the meaning and logic of the algorithm itself, a problem that can be overcome with visual block-based programming. For instance, during the workshops all the participants had some difficulties when learning about the main difference between *if-else* and *while* as a control structure when programming the robots to *make decisions* based on the data coming from the sensors. We noticed that the children felt very comfortable using the block-based programming environment of the Engino ERP as they could easily switch between different control structures (such as *if-else* and *while*) just by dragging and dropping the respective block elements to test different possibilities quickly and easily.

5.4. *The Potential of Informal Learning Environments for Developing CT Concepts and Practices in Children*

Programming is an activity that deals with abstract concepts and therefore one of the main challenges of teaching CT concepts and practices to people with little or no previous programming experience is to clarify misconceptions regarding computational concepts such as conditionals, loops, variables, and Boolean logic (Grover & Basu, 2017). Based on our findings it is possible to notice that reaching a full understanding of how to apply computational concepts and being able to successfully use control elements like loops and conditionals, associated with logical operators, can be a challenging process that takes time. Extracurricular activities like these workshops conducted at the public library can play a relevant role as a complement to formal education. Informal learning instances for developing CT and programming skills give children the chance to explore and test their computational creations in a friendly grade-free learning environment. It is, however, essential that the teaching activities are thoroughly designed so that the tutors will be able to use the computational tools they have (robots in this case) in a meaningful way and with a strong focus on constructionism. Also, the tutors must be able to explain,

both with words and by doing, the fundamental CT concepts and practices to avoid some misconceptions that tend to arise when teaching these abstract concepts to novice young programmers. Lastly, by continuously giving examples of robotic systems used in the real world, children become more motivated to learn how to program robots, as they see them as something more real and more meaningful, giving the learning experience another level of authenticity.

5.5. *Future Work*

In future studies we intend to explore in which way the programming interface of educational robots may influence how students understand CT and programming concepts, especially among novice programmers.

6. REFERENCES

Angevine, C., Cator, K., Roschelle, J., Thomas, S. A., Waite, C., & Weisgrau, J. (2017). *Computational Thinking for a Computational World*.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).

Carter, E. (2015). Its debug: Practical results. *Journal of Computing Sciences in Colleges*, 30(3), 9–15

Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. *Unpublished manuscript in progress, referenced in http://www. cs. cmu. edu/~ CompThink/resources/TheLinkWing. pdf*.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, *32*(6), 576-593.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education*, 63, 87–97.

Grover, S., & Basu, S. (2017, March). Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education* (pp. 267–272). Seattle, Washington: ACM.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38-43.

Grover, S., & Pea, R. (2018). Computational Thinking: A competency whose time has come. C*omputer science education: Perspectives on teaching and learning in school*, London: Bloomsbury Academic, 19-37.

Haduong, P., & Brennan, K. (2018, February). Getting unstuck: new resources for teaching debugging strategies in scratch. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 1092-1092).

Heintz, F., Mannila, L., Nordén, L. Å., Parnes, P., & Regnell, B. (2017, November). Introducing programming and digital competence in Swedish K-9 education. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 117-128). Springer, Cham.

Ker, C.L., Wadhwa B., Seow, P. S.K., & Looi, C.K. (2021). Bringing physical computing to an underserved community in an informal learning space. In C. K. Looi, B. Wadhwa, V. Dagiené, P. Seow, Y. H. Kee, & L. K. Wu (Eds.), *Proceedings of the 5th APSCE International Computational Thinking and STEM in Education Conference 2021* (pp. 101-106). Asia-Pacific Society for Computers in Education.

Kynigos, C., & Grizioti, M. (2018). Programming approaches to computational thinking: Integrating Turtle geometry, dynamic manipulation and 3D Space. *Informatics in Education*, *17*(2), 321-340.

Laurillard, D. (2013). *Teaching as a design science: Building pedagogical patterns for learning and technology*. Routledge.

Lu, J. J., & Fletcher, G. H. (2009, March). Thinking about computational thinking. In *Proceedings of the 40th ACM technical symposium on Computer science education* (pp. 260-264).

Mills, K., Coenraad, M., Ruiz, P., Burke, Q., & Weisgrau, J. (2021). *Computational Thinking for an Inclusive World: A Resource for Educators to Learn and Lead*. Digital Promise.

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers college record*, *108*(6), 1017-1054.

Mladenović, M., Boljat, I., & Žanko, Ž. (2018). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies*, *23*(4), 1483-1500.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas (1st Edition)*. New York, Basic Books.

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, *36*(2), 1-11.

Sanders, K., & McCartney, R. (2016). Threshold concepts in computing: Past, present, and future. *Proceedings of the 16th Koli Calling international conference on computing education research*, Finland.

Sullivan, A., & Bers, M. U. (2016). Robotics in the early childhood classroom: Learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education*, 26(1), 3–20.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, 49(3), 33-36.

Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, *6*, 20-23.

DOI: 10.34641/ctestem.2022.457

Modelling Zombies and Other Diseases

Faron MOLLER[1], Stewart POWELL[1], Phoebe ASPLIN[2], Dan ARCHAMBAULT[1],
Cagatay TURKAY[2], Graham MCNEILL[2], Max SONDAG[1,3]
[1]Swansea University, Wales
[2]University of Warwick, England
[3]University of Cologne, Germany
{F.G.Moller, S.W.Powell, D.W.Archambault}@swansea.ac.uk,
{Cagatay.Turkay, P.Asplin, Graham.McNeill}@warwick.ac.uk, sondag@cs.uni-koeln.de

ABSTRACT

Technocamps[1] is a national outreach programme based at Swansea University which – amongst many other things – provides STEM-based workshops to schools and young people. Before 2020, one popular set of Technocamps computational thinking workshops was on modelling the spread of diseases, which for our audience was a zombie infection. These workshops were all *unplugged* in nature and involved spreading diseases by passing around tokens. Different numbers of tokens were passed about, representing the ability of the disease to spread; and participants might be vaccinated giving them immunity. By changing such factors, the young people could watch how the disease might overcome the class or might die out. These workshops were particularly popular when presented in conjunction with the Royal Institution's Christmas Lectures – a series of popular lectures to young people – in December 2019 which were on understanding probability. Little did we know that our workshop series would take on a frighteningly real purpose a few months later. Throughout the COVID-19 pandemic, the public has been bombarded with messages about how governments were "following the science" and presented with images which "model the spread of the virus" and "track the R-value" in different regions of the world. Independent of our outreach activities, we developed visualization tools for public – and government – understanding of the science which we adapted into our outreach workshops. In this paper, we reflect on the effect of these workshops in explaining to young people the power of computational thinking in modelling diseases, and the extent to which they gained an understanding of this and of the current pandemic.

KEYWORDS

Computational thinking, modelling, pandemics, visualization, public understanding, outreach.

1. INTRODUCTION

The mathematical modelling of the spread of infectious diseases – e.g., HIV, influenza, or any number of tropical diseases – has long been of paramount importance. Public understanding of the field and its importance has historically been lacking, due in part to its success: whilst scientists have long been warning of the inevitability of global pandemics, by careful tracking and containment, infectious diseases have rarely progressed beyond the local endemic stage, and thus have stayed out of the public's attention.

There is one form of infectious disease, however, with which people in general have an obsession: zombieism. The 1968 film *The Night of the Living Dead* has cult status and continued popularity. It was already preceded by many such films, dating back at least to 1932's *White Zombie*, and zombies on screen retain a morbid appeal across the world's nations and cultures.

An article in *National Geographic* (Schriber, 2009) on the mathematical modelling of zombies being carried out by Canadian scientists (Munz et al, 2009) created a virus of its own. Within a week of publication, virtually every major news outlet in the world had picked up the story. Since then, there have been volumes of scientific articles devoted to the mathematical modelling of zombies (e.g., Smith, 2014).

Within a national school outreach programme, Technocamps, we deliver engaging workshops which develop young people's understanding of computational thinking as underpinning all STEM subjects. Our workshops are delivered to children of all ages and are typically unplugged. Modelling zombies has for some time been a popular topic for understanding mathematical modelling and in particular its computational aspects. The young people were learning by stealth something which they might never had imagined has real-world relevance, as they of course know that zombies are fiction. But in 2020, the COVID-19 pandemic suddenly made these workshops all too real.

Suddenly, the general public – and politicians – needed to know about and understand the science behind the mathematical modelling of COVID-19 and all of its variants. Within our scientific research programme, we developed a visualization tool which provides a simple interface for novice users to explore the spread of infectious diseases, in particular COVID-19. As it was intended to inform and be used by the general public, the tool was ideally suited to be adapted for use within our school workshops as an implementation of the unplugged activities.

In this paper, we describe the workshop that we have developed, both the unplugged activities and the use of the visualization tool; and provide an evaluation of its effectiveness as measured by feedback from the participants in the pilot sessions that have been carried out during its development. We conclude with a consideration of related work.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

[1] technocamps.ac.uk

2. BACKGROUND

2.1. *Technocamps*

Technocamps is a universities-based national schools and community outreach programme. It was founded at Swansea University in 2003, as a re-branding of the successful ITWales programme founded in 1993, but has since expanded to include a hub in every university in Wales. Its mission is to research, champion and deliver change in national curricula, qualifications, delivery, and professional development in order to foster a sustainable digital skills pipeline in Wales (Moller and Crick, 2018). It leads on national programmes of engagements in Wales with both primary and secondary school pupils and teachers, as well as adult learners through its Institute of Coding in Wales digital degree apprenticeship and micro-credential programmes.

A core Technocamps outreach activity is with school children. Since 2011, the programme has engaged with over 65,000 young people across Wales – roughly 7% of the Welsh population today aged 5-24, providing them with hands-on computational thinking workshops delivered in a fun, interactive and thought-provoking style that develops their intuitive understanding of the topics being delivered. Technocamps have developed and delivered a plethora of STEM-based workshops, covering a wide assortment of topics, ranging from the scientific (e.g., *Modelling Molecules*) through the social (e.g., *Technology, Ethics, and the Future*).

An important aspect of Technocamps is its support of teachers across Wales who are charged with teaching computational and digital technology subjects in school, given that only a quarter of them have any background in the subject (Moller & Powell, 2019). This has been made particularly challenging with the COVID-19 pandemic forcing teaching to be on-line and home-based. This has had a huge impact on children and their learning, which has motivated us to use the situation as a learning vehicle, not least to help children gain understanding of their "new normal".

2.2. *Visualisation Research*

The visualisation approach used within the workshops arises from a project led by visualisation and epidemiology researchers from Swansea University and the University of Warwick[2], as well as collaborations developed through the Scottish COVID-19 Response Consortium[3] (Chen et al, 2022). In addition to developing visual analytics methods for experts in analysing large collections of contact tracing networks, our aims include developing methods for the communication of disease models to broader audiences in transparent, comprehensive, and engaging ways. The visualisation tool that was used to facilitate the workshops discussed in this paper were driven by this latter ambition.

The design and development of the visualisation tool have been informed by a two-year collaboration. We started by exploring how computational simulations of disease progression could be visually analysed to improve the simulations, and to assess different contact-tracing strategies (Sondag et al, 2022). As expected, these computational epidemiological models are complex and involve several parameters which makes it challenging to communicate their results to broader audiences. We thus developed a web-based, agent-based simulator based on a simpler model that approximates this behaviour[4], as well as an accompanying interactive visualisation framework to be used in engagement and training activities[5]. The visualisation tool described in this paper is a case study where the web-based simulator and the visualisation framework have been applied within the context of disease progression through contact.

3. THE ZOMBIE WORKSHOP

As with most of our workshops, the infectious disease modelling workshop is designed to be delivered within a classroom-style environment, to 25-30 young people aged between 8 to 16, over the course of a morning or an afternoon. To make it lively and engaging, the premise of the workshop is a zombie outbreak that is spreading amongst the participants of the workshop.

The workshop incorporates three modes of delivery. Firstly, a standard presentation style with multimedia resources is used to convey information and engage the participants in discussion. Then a series of unplugged activities are carried out to help participants understand the ways in which infectious diseases spread, and how their spread can be modelled computationally. Finally, a visualization tool is introduced with which the participants can interact in order to understand how different infection rates and varying preventive measures affect the spread of the disease.

3.1. *Unplugged Activities*

The workshop begins by introducing the concept of *state* in order to track the zombie infestation as it propagates through the classroom. We start with two basic states: a green state to represent an uninfected member of the class; and a purple state to represent an infected member of the class; i.e., the zombies. We then introduce the notion of a simple contagion *process* whereby, at any given point in time, an infected member of the class can infect a nearby classmate.

Figure 1. A Workshop in Action

[2] contact-viz.cim.warwick.ac.uk

[3] www.gla.ac.uk/research/az/scrc

[4] gjmcn.github.io/atomic-agents

[5] gjmcn.github.io/atomic-agents-vis

This forms the basis of our first unplugged activity. Using string to represent connections between two participants, and the flip of a coin to decide whether a zombie infects a neighbouring participant, the class tracks the spread of the zombie apocalypse. In some scenarios, the whole class becomes infected almost immediately; and in others, the zombies take much longer to infect everyone. The string is useful for investigating the history of an infection, from one infected person back to "patient zero".

This first activity inevitably raises several important questions about the spread of diseases. Can a person recover from the disease? Can a person die from the disease? Is there any known vaccination against the disease? With these questions in mind, new states are introduced: a blue state to represent a member of the class who has recovered from or is immune to the disease; and a yellow state to represent a zombie that has died. The participants then re-run the process and notice the differences that these new states create, as well as the added complexity of tracking them.

After exploring this on paper and trialing it several times in animated form, attention is brought to the idea of how this could be a way of understanding how real-life infectious diseases spread in the real world. It is quickly agreed that you would need to be able to track the state of each participant – which even with a small number is quite complex – and that you would need a lot more participants. The conversation is turned towards using a computer to do this, which is when the visualization tool is introduced.

3.2. Visualization Tool

The visualization tool we devised provides a simple interface for participants to explore the spread of an infectious disease, using a similar style to those introduced in the unplugged activities. The ability to quickly run simulations and understand the effects of manipulating the probabilities of infection and other parameters provides understanding of how infectious diseases spread and how a computational model can be used to predict their spread.

The simulation is first explained to the participants so that they can associate it with the unplugged activities they have just completed. Each person is represented by a dot within the model. Each dot has a particular state that is represented by a color: light blue meaning healthy; red meaning sick; dark-blue meaning recovered; and yellow meaning dead. The model also introduces the concept of *exposure radius* – the distance at which dots will spread the disease to – which is represented by light grey circles.

Initially, participants are challenged to run the simulation using default settings and investigate the different outcomes produced by running the model several times. This is an important step for participants to reinforce the randomness of these simulations, noting that the simulations will run and evolve differently each time.

Additional functionality of the tool is then explained to the participants, bit-by-bit. Different simulation parameters are explained which can be tweaked in order to affect the speed and effectiveness of the spreading of the infectious disease. These are: the probability that a dot becomes sick when exposed; the probability that a dot recovers or dies; and the exposure radius.

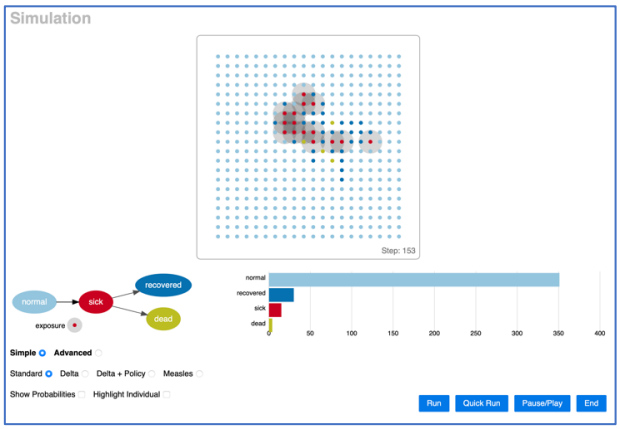


Figure 2. The Visualization Tool. Participants see an infectious disease spread by noting the changing of states (colours) of the dots (people). There is a model diagram explaining the state changes, and a bar chart showing the number in each state at any given time.

Such parameters are associated with every infectious disease. Users can set these parameters themselves or opt for pre-set parameters for existing diseases – e.g., for various COVID variants or for measles – and consider the relative dangers of different diseases. By playing around with different scenarios, participants gain an understanding of the direct link between the model and the COVID-19 pandemic.

The tool then allows for further explorations in which vaccinations can be introduced, and the movement of the dots can be restricted to between home and school or office, or even totally restricted to represent a lockdown scenario. Future developments involve representations of interventions relating to social distancing, social bubbles, and limits on social gatherings; as well as the effect of varying degrees of adherence by the public on such restrictions, in order to explore the effect of people not obeying social distancing and other rules and guidelines that are introduced by their government.

4. EVALUATION

Pilot workshops were delivered to 140 young people from the ages of 9 to 15, in both classrooms and in science festivals. Data was collected from participants using pre-and post-workshop questionnaires that were completed on the day of the workshop. Quantitative data was collected using a 5-point Likert scale and qualitative data by providing participants with an open space to provide their answers.

During the pre-questionnaire, we asked participants to rate and comment on their understanding of how different diseases spread. Of the 137 participants that answered this question, 91 (66%) rated their understanding as nothing (1) to unsure (3). The qualitative data showed a clear trend of participants associating coughing, sneezing and close contact as the primary cause of a disease spreading. In the post-questionnaire, when asked about how much they had learnt about how different diseases spread, of the 123 participants that answered this question, 99 (80%) rated that they had learnt somewhat (4) to a lot (5).

We also evaluated the delivery method of the workshop and asked participants which part of the workshop they felt that

they learnt the most from: the presentation, the physical (unplugged) activities, or the visualization tool. Although not the initial intention, a number (30) of participants selected more than one choice. These have been coded against both choices in the table in Figure 3.

	Presentation	Physical Activities	Visualization Tool
All	36	33	63
Aged 9-10	17	14	33
Aged 11-15	19	19	30

Figure 3. Feedback on Effective Learning Modes

Whilst the unplugged activities were clearly appreciated and made the workshops engaging, the use of the visualization tool was clearly what the participants felt they learnt the most from. Initially, based on feedback from initial workshops held at a science festival, we had predicted that younger students would prefer and gain more from the unplugged activities in comparison to the use of the visualization tool; however, this did not turn out to be the case with the school-based workshops. The reason for this, we presume, is due to our own refinement of how the tool is presented and used in the workshops, which evolved from the early science festival presentations in which the visualization tool was used as an add-on to our established fun and engaging interactive workshop.

5. CONCLUSIONS AND RELATED WORK

School children have been hugely affected by the COVID-19 pandemic, and engaging with them in workshops which directly address the cause of the pandemic is warranted. There is evidence that greater knowledge and understanding by young people of infectious diseases leads to more positive attitudes and healthier behaviour (Myant & Williams, 2005). Efforts have been made to impart onto young people understanding of the *biology* of viruses, and in particular COVID-19 (Manches & Ainsworth, 2022). Our efforts are aimed at providing an understanding of the mathematical and computational modelling of COVID-19 and related infections, as these are the bases of the messages they receive and which directly affect their expected behaviour.

Interactive visualisations have previously been developed for similar purposes, an example being the interactive model provided by Stanford University (Childs et al, 2020). Our aim has been to simplify the assumptions and make our visualisations graphically appealing to young people in order to fully engage them.

Another common way to be engaging is through gamification. For example, in the *Can You Save The World* game (Wiseman & Martin, 2020), players gain or lose points depending on their behaviour whilst walking around a virtual environment. This is particularly suited to younger children, but lacks the scope for understanding the effect on whole communities. Our approach is to embody this gamification in our physical unplugged activities, which we find more effective than – and at least as fun as – employing a single-player on-line game.

6. REFERENCES

Chen, M., Abdul-Rahman, A., Archambault, D., Dykes, J , Slingsby, A. , Rtisos, P. D., Torsney-Weir, T., Turkay, C., Bach, B., Borgo, R., Brett, A., Fang, H., Jianu, R. , Khan, S., Laramee, R. S., Nguyen, P. H., Reeve, R., Roberts, J., Vidal, F., Wang, Q., Wood, J., & Xu, K. (2022). RAMPVIS: Answering the Challenges of Building Visualisation Capabilities for Large-scale Emergency Responses. *Epidemics*.

Childs, M., Kain, M., Kirk, D., Harris, M., Ritchie, J., Couper, L., Delwel, I., Nova, N., & Morecai, E. (2020). Potential Long-Term Intervention Strategies for COVID-19. Available at: http://covid-measures.stanford.edu (Accessed May 2, 2022.)

Manches, A., & Ainsworth, S. (2022). Learning About Viruses: Representing COVID-19. *Frontiers in Education, Volume 6*:736744.

Moller, F., & Crick, T. (2018). A University-based Model for Supporting Computer Science Curriculum Reform. *Journal of Computers in Education* 5:415-434.

Moller F, & Powell, S. (2019). Technoteach: Supporting Computing Teachers Across Wales. *Proceedings of WiPSCE'19: the 14th Workshop in Primary and Secondary Computing Education*, Article No. 9, ACM Press.

Munz, P., Hudea, I., Imad, J., & Smith, R. (2009). When zombies attack: Mathematical modelling of an outbreak of zombie infection. In *Infectious Disease Modelling Research Progress*, J.M. Tchuenche & C. Chiyaka (eds), Nova Science, Happuage, NY, 133-156.

Myant, K. A., & Williams, J. M. (2005). Children's Concepts of Health and Illness: Understanding of Contagious Illnesses, Non-contagious Illnesses and Injuries. J. Health Psychol. 10 (6), 805–819.

Scriber, B. (2009). What Can Zombies Teach Us About Swine Flu? *National Geographic,* 12 August 2009.

Smith, R. (2014). *Mathematical Modelling of Zombies.* University of Ottawa Press.

Sondag, M., Turkay, C., Xu, K., Matthews, L., Mohr S., & Archambault D. (2022). Visual Analytics of Contact Tracing Policy Simulations During an Emergency Response, *Computer Graphics Forum*.

Wiseman, R., & Martin, J. (2020). Can You Save the World? Available at: https://martin-jacob.itch.io/can-you-save-the-world (Retrieved May 5th, 2021).

 DOI: 10.34641/ctestem.2022.458

Integrating Game-based Learning into Computational Thinking Class for Lower Primary Students: Lesson Design and Course Effect

Shuhan ZHANG[1*], Gary K. W. WONG[2], Peter C. F. CHAN[3]
[1,2] Faculty of Education, The University of Hong Kong, Hong Kong
[3] NetDragon, Hong Kong
shuhan@connect.hku.hk, wongkwg@hku.hk, peterchan@elm-tree.com

ABSTRACT

Computational thinking (CT) has been integrated into K-12 curricula globally. With the growing trend of initiating CT in early childhood education, great effort has been made in developing age-appropriate courses targeting young children. This study aims to introduce an instructional unit of CT instruction for children aged 5-7, Coding Galaxy-Foundation, where unplugged activities and digital game-based learning were applied. A public primary school in Hong Kong was invited for delivering the course, where Grade 1 and Grade 3 students were involved (N=57). Six lessons were selected, covering basic CT concepts including sequences, decomposition, events, relative direction, debugging, loops, pattern recognition, and conditionals. Each lesson consisted of three sections, namely, a) concept introduction with daily-life examples, b) unplugged activities based on puzzles, and c) digital game practices with Coding Galaxy game app. Students' attainment from the course was assessed in both cognitive and attitudinal facets. The results indicated that the course was effective in sustaining students' CT cognitive performance and improving students' coding attitudes, and female and Grade 3 cohorts were the most beneficiaries. Implications for further research and educational practices are discussed.

KEYWORDS

Computational thinking, early childhood education, unplugged activities, digital game-based learning, coding

1. INTRODUCTION

In the digitized society, computing skills have become increasingly important for citizens. In recent years, computational thinking (CT), the major subset of computing, has been integrated into K-12 education to support the future development of young children. CT stemmed from "algorithmic thinking" raised by Seymour Papert in 1980, who describes that it is "the art of deliberately thinking like a computer, according, for example to the stereotype of a computer program that proceeds in a step-by-step, literal, mechanical fashion" (Papert, 1980, p.27). Later, the phrase "computational thinking" became well-known after Jeanette Wing re-proposed the notion in 2006, depicting it as "an approach to solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006, p. 33).

Under Wing's call, CT has been integrated into K-12 curricula on a global scope (Bocconi et al., 2016). More recently, heat can be seen in initiating CT education for younger students (Bers et al., 2018). It was reported that CT education has permeated into elementary school classrooms (Rich et al., 2019) and even kindergartners as young as 4-year-olds could acquire basic CT concepts (Strawhacker et al., 2018). The growing trend of implementing CT in early childhood education necessitates the development of age-appropriate instructional applications to support student learning.

To support CT education, substantive attempts have been made in designing manifold learning tools, encompassing unplugged activities, block-based coding platforms, robotics, and digital coding games (Shute et al., 2017). Among these initiatives, unplugged learning activities and digital games tend to be child-friendly. Unplugged activity is an approach to teaching coding without digital devices but with tangible objects instead (e.g., cards, puzzles) (Bell et al., 2009), which is suitable for young novices to start with (Zhang & Cui, 2021). Digital coding game, on the other hand, offers an entertaining playground to support interactive coding practices (Giannakoulas & Xinogalos, 2018). The platform embraces goal-oriented tasks and instant progression feedback, enabling a low entry bar for young children (Zhang et al., 2021).

Since the outbreak of COVID-19, the world has been going through dynamic changes. In the educational field, the biggest influence lay in the teaching mode, where physical face-to-face instructions have been hindered. Thus, technology-based distance learning has become the main solution to the schooling system (Daniel, 2020). The situation has driven us to explore proper instructional approaches to allow student-centered learning practices. For CT education, due to the portability of tangible materials and the low entry bar of entertaining games, unplugged activities and digital coding games appear to be highly applicable in distance learning and individual practices in home-based settings.

This study aims to introduce an instructional unit implemented during COVID-19, where unplugged activities and digital game-based learning approach were applied in a CT course designed for children aged 5-7. We intend to explore the effectiveness of the course in both cognitive and attitudinal aspects and offer suggestions for CT early childhood education. The study is guided by the following research questions.

RQ1: What was the effect of the course on students' CT cognitive performance?

RQ2: What was the effect of the course on students' coding attitudes?

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

2. METHOD

2.1. Participants

Through convenient sampling, a public primary school in Hong Kong was invited to deliver the course. As the course targeted lower primary students, pupils from Grade 1-3 were invited, and a total of 57 students agreed to participate, comprising of 1st and 3rd graders. Note that this cohort did not have any CT or coding learning experience at school before the intervention, and thus they were considered novices in coding and CT.

2.2. *Lesson Design*

The Coding Galaxy-Foundation course (CG-Foundation) was used. CG-Foundation aims at introducing CT and basic coding concepts to young children, aged 5–7, without any prior experience in coding. Six chapters from the course were taken to conduct six lessons (see Table 1). The course activities and explanations involve age-appropriate everyday examples that children can relate to, emphasizing that the concepts apply to daily problem solving as well as computer coding. In addition to using everyday examples, some activities also connect CT to other school subjects, such as reading and mathematics. The course focuses on exploring CT and coding concepts for problem-solving, instead of using a specific coding tool to learn to program.

Comprehensive teaching materials were provided to teachers to conduct each lesson, highlighting the concepts, objectives, and details for each activity. Each lesson contained three sections, namely, a) concept introduction, b) unplugged activities, and c) digital game practices. First, a CT concept was elaborated with daily life examples (Figure 1). Then, unplugged activities were assigned, embracing board games where students could manipulate tangible objects (e.g., cards) to complete the tasks based on the taught concept (see Figure 2). Students need to identify viable routes and use the available cards to work out the solutions. Lastly, the CG game app was applied (see Figure 3), which contained coding puzzles corresponding to the CT concepts in the lessons. In the game, players can control the character to lead him to the destination with simple coding languages. For each task, related coding commands for the solution were provided, which can be added to the panel on the right through drag and drop.

The overall structure of the game was designed to focus on introducing the topics one at a time, providing a progression from easy to difficult puzzles with one or more concepts involved. Successful completion of the puzzles involved the correct application of the concepts, and the player would be rewarded in the game with stars (see Figure 4). Three stars were awarded for the optimal solution, where an accurate route was executed with the fewest commands while collecting all the crystals. Two stars were awarded for identifying a semi-optimal route or missing collecting some of the crystals. One star was awarded to those who use an inefficient route to the destination, implying a lack of pattern recognition and path optimization skill.

*Table 1.*Selected Lessons and Descriptions.

Lesson	Key concept	Description
1	*Sequences, Decomposition*	Learn the importance of sequences in doing things through decomposing everyday examples, giving clear instructions.
2	*Sequences, Events*	Learn that “events” trigger responses, and reflect on surrounding observations of applications of “events”.
3	*Sequences, Relative direction*	Arrange sequential instructions for routes using relative direction commands, relating to the concept of direction in mathematics.
4	*Debugging*	Find and correct errors in existing instructions.
5	*Loops, Pattern recognition*	Identify patterns in problems, and use loops in setting up solutions.
6	*Conditionals*	Identify conditionals in daily life to make decisions and plans.

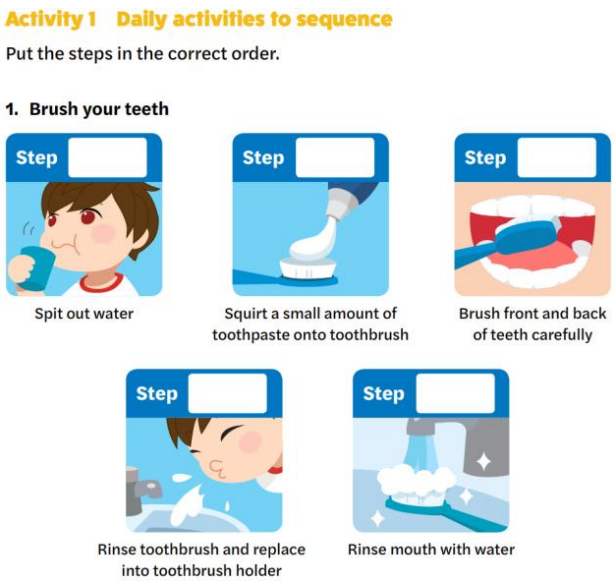


Figure 1. Activity based on Daily Examples.

Figure 2. Unplugged Activities.

Figure 3. CG Game-Puzzle.

Figure 4. CG Game-Rewarding Page.

2.3. Measurements

Two instruments were adopted for this study. The instrument for CT cognitive performance was adapted from Computational Thinking Test for Lower Primary (CTtLP, Zhang et al., 2021). The target age group of CTtLP was 6-9, which is suitable for our participants. Eighteen items were selected, covering the relevant CT concepts of the course. Proper psychometric properties were yielded in the original study in the Chinese context.

For assessing students' coding attitudes, Elementary Student Coding Attitudes Survey (ESCAS) (Mason & Rich, 2020) was adopted. Three factors were selected from the scale, namely, coding confidence, coding interest, and coding utility. The Chinese version of the scale, ESCAS (Chinese), was validated (Zhang et al., 2022), yielding adequate psychometric evidence. Thus, ESCAS (Chinese) was adopted.

2.4. Procedure

Six lessons were administered, covering the six chapters in Table 1. The course was delivered online via ZOOM, due to the pandemic situation. In the first and last lesson, apart from teaching, a pretest and post-test were delivered respectively, consisting of CTtLP and ESCAS (Chinese).

3. RESULT

Due to the age differences, the test results for each grade were extracted, with 27 1st graders and 30 3rd graders, and findings will be reported by grade levels.

3.1. RQ1: Effect on CT Cognitive Performance

The effect on CT cognitive performance was analyzed with paired sample t-test, and the bar chart is presented in Figure 5. For Grade 1, scores decreased slightly from 10.48 (SD=6.10) to 9.37 (SD=4.87), yet not significant (p=0.31). Grade 3 performed better than their Grade 1 counterparts, having a mean of around 11 for both tests, with 11.53 (SD=5.22) and 11.03 (SD=5.42) respectively, while no difference was detected between the tests (p=0.47). This indicates that students' CT cognitive performance remained stable for both grades.

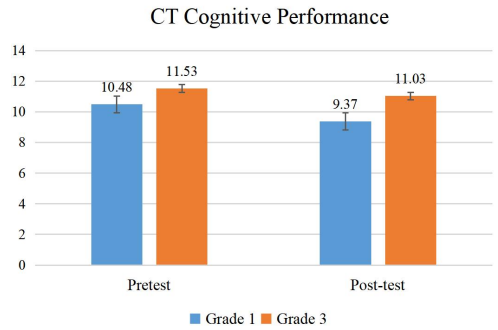


Figure 5. Bar chart of CT performance scores.

3.2. RQ2: Effect on Coding Attitudes

Coding attitudes were measured in three facets, namely, coding confidence, coding interest, and coding utility. Paired sample t-test was used to examine the change in the three dimensions before and after the intervention. Results for coding confidence are displayed in Figure 6. For both grades, an improvement could be seen between pretest and post-test, with 3rd graders yielding a bigger increase from 4.32 (SD=1.44) to 4.70 (SD=1.17). Regarding coding interest (see Figure 7), while no obvious change was detected for 1st graders, an upward trend was observed for 3rd graders, climbing moderately from 4.45 (SD=1.55) to 4.85 (SD=1.13). Similar findings were generated from coding utility (see Figure 8), where Grade 3 students saw significant growth from 4.20 (SD=1.42) to 4.91 (SD=1.21) ($p<0.05$). The results illustrate that Grade 3 students seemed to benefit more from the course, with observed improvement in multiple attitudinal facets towards coding.

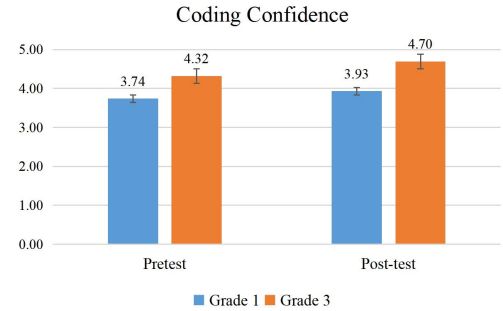


Figure 6. Bar chart of Coding Confidence.

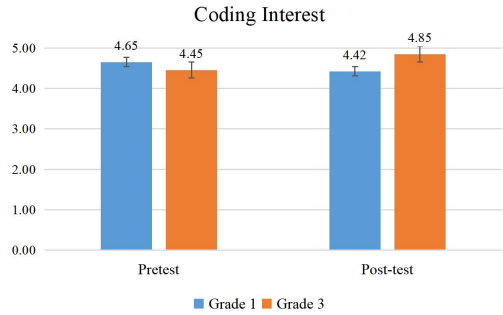


Figure 7. Bar chart of Coding Interest.

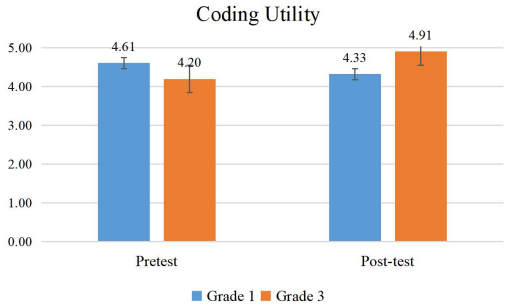


Figure 8. Bar chart of CT Coding Utility.

Further, we identified the pre-post changes across gender groups. While neither gender differences nor apparent changes were seen for coding interest and coding utility, unexpected results were found regarding coding confidence. Figure 9 displays the bar chart of coding confidence, illustrating that there was no difference between gender groups in the pretest, with both groups reporting a score of around 4, whereas in the post-test, females rated their confidence much higher (N=27, mean=4.46, SD=1.31), even surpassing their male counterparts (N=30, mean=4.22, SD=1.28). The results imply that girls might be greater beneficiaries of the course for gaining more confidence in coding afterward.

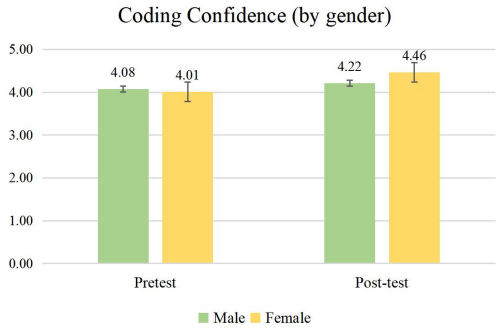


Figure 9. Bar chart of Coding Confidence by Gender.

4. DISCUSSION AND CONCLUSION

This study introduced an instructional unit of a CT course designed for young children aged 5-7, CG-Foundation, where unplugged activities and digital game-based learning approach were applied. Six lessons were taught to Grade 1 and Grade 3 students in a public primary school in Hong Kong. Each lesson was composed of concept introduction, unplugged activities, and digital game practices. Due to the pandemic situation, CG-Foundation was delivered online. The instructional design of the lessons allows for distance teaching, as the portability of unplugged materials and playfulness of digital games enabled student-centered learning practices.

The effectiveness of the course was examined regarding cognitive attachments and attitude change. Results indicate that students' CT performance remained constant. This is understandable under the pandemic case, where children's learning efficiency may be influenced in a home-based learning context. Also, six weekly sessions may be too short to cause dramatic growth. On the other hand, coding attitudes yielded positive results, with female and Grade 3 students being the main beneficiaries. Girls owned more confidence in coding after taking the course, and 3rd graders saw a significant improvement in coding utility and moderate increments in coding confidence and coding utility. Overall, CG-Foundation is an applicable course for lower primary students to learn basic CT concepts, which is viable for both physical and distance learning contexts.

5. REFERENCES

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, *13*(1), 20-29.

Bers, M. U. (2018). Coding and computational thinking in early childhood: the impact of ScratchJr in Europe. *European Journal of STEM Education*, *3*(3), 8.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing computational thinking in compulsory education. *European Commission, JRC Science for Policy Report*, *68*.

Daniel, J. (2020). Education and the COVID-19 pandemic. *Prospects*, *49*(1), 91-96.

Giannakoulas, A., & Xinogalos, S. (2018). A pilot study on the effectiveness and acceptance of an educational game for teaching programming concepts to primary school students. *Education and Information Technologies*, *23*(5), 2029-2052.

Papert, S. (1980). Children, computers and powerful ideas. In. New York: Basic Books.

Rich, P. J., Hu, H., Christensen, J., & Ellsworth, J. (2019). The Landscape of Computing Education in Utah. *Grantee Submission*.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158.

Strawhacker, A., Lee, M., & Bers, M. U. (2018). Teaching tools, teachers' rules: exploring the impact of teaching styles on young children's programming knowledge in ScratchJr. *International Journal of Technology and Design Education*, *28*(2), 347-376.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Zhang, S., & Cui, C. (2021). Implementing Blended Learning in K-12 Programming Course: Lesson Design and Student Feedback. *Proceedings of 2021 IEEE Integrated STEM Education Conference (ISEC)*.

Zhang, S., Wong, G. K. W., Chan, P. C. F. (2021). Achievement and Effort in Acquiring Computational Thinking Concepts: A log-based Analysis in a Game-based Learning Environment. *Proceedings of Fifth APSCE International Conference on Computational Thinking and STEM Education 2021 (CTE-STEM)*.

Zhang, S., Wong, G. K. W., & Sun, X. (2022). Exploring Coding Attitudes of Chinese Elementary Students: A preliminary study. *Proceedings of 2022 IEEE Integrated STEM Education Conference (ISEC)*.

DOI: 10.34641/ctestem.2022.460

Integrating CT into Biology: Using Decision Tree Models to Classify Cell Types

Jacqueline NIJENHUIS-VOOGT *
Radboud University
The Netherlands
jacqueline.nijenhuis@ru.nl

Sabiha YENI
Leiden University
The Netherlands
s.yeni@liacs.leidenuniv.nl

Erik BARENDSEN
Radboud University & Open University
The Netherlands
erik.barendsen@ru.nl

ABSTRACT

The integration of computational thinking (CT) into subject learning has the potential not only to foster digital literacy, but also to deepen STEM learning because the use of computational models and development of computational solutions advances students' understanding of subject area content. However, designing and implementing a curriculum that effectively integrates STEM and CT is challenging for educators because they have little experience in computing terminology, key concepts, and approaches to learning. We therefore aimed to develop CT integrated K-12 lessons in collaboration with subject teachers to determine suitable CT learning objectives as well as teaching and learning strategies. In this study, we focus on a 9th-grade biology lesson where students were asked to construct decision trees for determining cell types in as few steps as possible. Decision trees form a computational model that fits a wide range of classification systems in biology. We investigated the effect of using decision trees on students' biology and CT learning outcomes by analyzing their end products in the assignment. Additionally, we analyzed students' and teachers' views about the CT integrated lesson using questionnaires and semi-structured interviews. We found that students valued the way a decision tree helps them to structure the information. The teacher expressed that drawing a decision tree enabled the students to reason about the cell types, fostering a different way of thinking. Regarding CT, decision trees may help to improve decision analysis and classification, which are related to abstraction and algorithmic thinking skills.

KEYWORDS

Computational thinking, STEM, biology, K-12, decision tree.

1. INTRODUCTION

1.1. CT Integration into STEM Disciplines

In an effort to deepen learning in K-12 Science, Technology, Engineering, and Mathematics (STEM) fields, there is an increasing interest to integrate computational thinking (CT) into subject learning. Computational approaches are vital for STEM practices because how these disciplines are practiced in the professional world is rapidly changing (Foster, 2006). In recent years, STEM fields have been supported with computational practices, for example in Bioinformatics, Computational Statistics, Chemometrics and Neuroinformatics (Weintrop et al., 2016). Bringing computational tools and practices into K-12 mathematics and science classrooms gives learners a more realistic view of what these fields are, and better prepares students for professional careers in these disciplines (Augustine, 2005). This sense of authenticity and real-world applicability is important in the effort to motivate diverse and meaningful participation in computational and scientific activities (Blikstein, 2013; Weintrop et al., 2016).

There are many research studies about new learning environments, tools and activities designed to promote computational thinking skills in different science contexts. In these studies, several science topics in K-12 are presented in which CT may be embedded. For example: simple electronic circuit, circuit diagram (Jacobson et al., 2015; Kafai et al., 2014), digital and analog waves, wave amplitude and frequency, modern sonography (Lehmkuhl-Dakhwe, 2018); kinematics (Basu et al., 2015); geology, meteorology, astronomy, and energy (Peel et al., 2015). There are also efforts to include different media and tools in science such as programming environments such as Scratch (Resnick et al., 2009) and Alice (Lee et al., 2011); computational modeling environments such as NetLogo (Wilensky and Rand, 2015); electronic prototyping kits such as Arduino and digital textiles (Buechley et al., 2008); video games including Quest Atlantis (Barab et al., 2005) and RoboBuilder (Weintrop & Wilensky, 2013).
Many investigations focused on the impact of programming skills or computational media towards learning computational thinking (CT). However, not all teachers are able to implement or teach a programming curriculum at the K-12 level. In this sense, another notable approach to bringing computational thinking into K-12 classrooms is the use of unplugged activities (i.e., in which there is no use of digital devices). In this study, we investigated the use of decision trees as an unplugged approach to enhance biology and computational thinking skills of 9th grade students.

1.2. Decision Tree Models' Relation with CT

A decision tree is a decision support tool that uses a flowchart-like model/diagram of decisions and their possible consequences to help identify a strategy most likely to reach a goal. It is one way to display an algorithm that only contains (possibly nested) conditional control statements (if-then-else). The structure of a decision tree is built on three main parts: a root node, decision nodes (or branches) and leaf nodes (Figure 1).

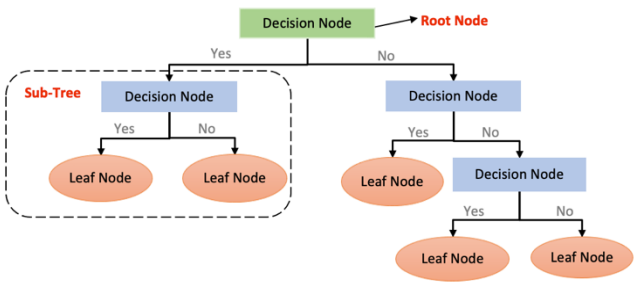


Figure 1. Decision tree structure

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

The root node is the starting point of the tree, and both root and decision nodes contain questions or criteria to be answered. Branches are arrows connecting nodes, showing the flow from question to answer. Each node typically has two or more nodes extending from it. For example, if the question in the first node requires a "yes" or "no" answer, there will be one node for a "yes" response, and another node for "no." Leaf nodes represent the classifications. Decision trees are commonly used in decision analysis and classification; they are also a useful tool in machine learning (Othman et al., 2018).

A decision tree model may help to improve CT skills because it is a model that fits a wide range of classification systems in biology. By using or drawing a decision tree, students can visualize the relationship between the characteristics of a subject related phenomenon. Designing simple algorithms in the form of decision trees could assist in the strengthening of algorithmic thinking skills and can show that there are different algorithms to reach the solution. Furthermore, students are supposed to separate important from redundant data while creating their decision nodes which may contribute to enhance students' abstraction skills. While designing decision trees, students are supposed to divide a larger and more complex task (root node) into several sub-tasks (decision nodes) which is related to students' decomposition skills. It also seems a helpful tool for improving evaluation skills of students, because they can evaluate the decision trees according to predefined criteria and they can see the quality of their solutions and make improvements. For teachers, it can be used as a formative assessment tool because students' understanding and misconceptions can be easily noticed in decision trees and the evaluation of a decision tree can be automated (Petrović & Pale, 2017).

1.3. Decision Tree Applications in Biology

Bioinformatics is a growing scientific field created by the intersection of biology, computer science, and information technology to support the storage, organization, and retrieval of biological data (Wheeler et al., 2006). It is important to know how to support secondary school students to engage with real-world science developments using scientific and computational techniques, such as decision trees. Decision tree approaches have been shown to have wide applications with high performance in solving bioinformatics problems. For example, there have been several attempts to use decision trees for the classification analysis of the gene expression data (Dudoit & Fridlyand, 2002). Specifically, decision tree approaches have been widely applied in cancer classification and in annotating multilevel genomic sequences (Che et al., 2011; Salzberg et al., 1998). Bioinformatics can therefore be used as a context to introduce students to real-world environmental datasets and to support students in developing their CT skills.

The real-world applicability of decision trees is important for motivating teachers' and students' participation in computational and scientific activities. Another motivation for this research study is that there is little guidance and support for science teachers to integrate CT with existing content (Grover & Pea, 2013; Weintrop et al., 2016). For this reason, the aim of this study is to develop a CT integrated biology lesson plan together with a science teacher and to investigate the effect of this lesson and the decision tree applications on 9th grade students' biology and CT skills while learning about cell types. Also, we aim to explore the students' and teacher's attitudes towards a CT integrated lesson and their views about the lesson. Consequently, this study addresses the following research questions: 1) What are the biology and CT related learning outcomes of a CT integrated lesson? 2) What are the students' and teacher's attitudes towards a CT integrated lesson?

2. METHODOLOGY

This study is part of a larger project (Barendsen, 2022) focusing on the definition of learning trajectories for CT integration into the K-12 curriculum. In particular, this study focused on investigating the effectiveness of a CT integrated biology lesson. To this end, we employed a qualitative case study approach (Stake, 1994) to explore the effectiveness of a CT integrated lesson (specifically decision trees) on high school students' biology and CT skills while learning about cell types. Patton (2002) defined cases as a "specific, unique, bounded system… [in which researchers] gather comprehensive, systematic, and in-depth information" (p. 447). Within this approach, the CT integrated biology lesson is our specific unit of interest. The following (sub-)sections describe the research design, participants, data collection, and data analysis process.

2.1. Lesson Design

This case study was implemented in a secondary school in the Netherlands. The participating teacher attended a workshop where he was shown CT concepts and examples of lessons in which CT was integrated into several disciplines across the curriculum, such as science, humanities, and languages. Then, the teacher worked with the researchers individually and developed a lesson plan about a biology topic he planned to teach anyway, as well as with the level and type of CT he felt comfortable with. In this biology lesson, students learn about the cell types. Learning objectives of the lesson are presented in Table 1.

Table 1. Learning Objectives (LO)

Subject-related LO	Computational Thinking-related LO
Identify different types of cells	Use diagrams to represent data at an abstract level (AB)
Describe the characteristics of each type of cell	Separate the important from the redundant information (AB)
Use the cell type knowledge to create a decision tree	Design simple algorithms such as if statements for decisions (AT)
	Recognize that different algorithms exist for the same problem (AT)
	Use criteria to evaluate the quality of solutions and identify improvements (EV)
	Check whether no important part is missing or forgotten when performing partial assignments (EV)

AB: Abstraction, AT: Algorithmic Thinking, EV: Evaluation

There are three different instructional strategies applied in this lesson plan: direct instruction, scaffolding, and collaborative learning. At the beginning of the lesson, the teacher used direct instruction to summarize the topic of cells and kingdoms and the types of cells. They had already discussed this topic in the previous lesson. At the start of the lesson, students were sitting in a circle and it allowed everyone to see each other and helped to promote engagement. Then the teacher shared the printouts with the

assignments. The teacher explained that students were going to construct a decision tree to determine the types of cells (bacteria, fungi, plants, and animals) in as few steps as possible. He described decision trees and the way students could draw them. Students were encouraged to ask their questions. They moved to their desks in groups and drew their decision trees on paper individually. Scaffolding was implemented by the teacher to give support to students whenever needed. As a part of collaborative learning, students had their decision tree checked by a classmate. They evaluated their classmate's decision tree according to predefined criteria (content, classification, and presentation/visualization). Then, they used the feedback obtained to create a second version of their decision tree.

2.2. Participants

In total, 22 9th grade students were involved in this study. For 18 students, we received signed consent forms. The age range of students is between 13 and 16, with an average of 14. The participants were nine girls and eight boys; one participant reported the gender as other. All students (except one student) have either a computer, a tablet, or a smartphone available if they need them. They were also asked about their programming experience. Prior activities in school that focused on CT were related to programming. Six students have never taken a programming lesson. 10 students have had a programming lesson, however six of them have had lessons for less than one month and three of these 10 students have had programming lessons less than one year. Only one student has had programming lessons for 2-3 years. The most used programming language is Scratch, which was reported by five students. They were also asked to rate their programming experience between one (no experience) and five (very experienced). Half of the students (nine) rated themselves as one or two, five students rated as three and only two students rated themselves as four or five. 12 students reported that they need help during programming. In previous biology lessons, students have worked with models that resemble decision trees, for example classification charts. The biology teacher is 34 years old, male, and has 14 years of teaching experience. He is a project leader at digital literacy projects and attended this research voluntarily. He has no prior programming experience.

2.3. Data Collection and Analysis

The data were gathered using a short survey, an exit ticket, an artifact/end product (decision tree) and interviews with a few students and the teacher. The short survey included questions on age, gender, grade, programming experience and self-efficacy. The exit ticket was completed at the end of the lesson, to understand students' attitude toward a CT integrated biology lesson. It includes questions about various topics (enjoyment, interest, clarity, comprehension, difficulty) with a three-point Likert scale and eight open-ended questions about attitudes toward lessons, likes/dislikes etc. Students' decision trees and their classmates' feedback were collected. Additionally, four students were interviewed and were asked to elaborate on some of the questions from the questionnaire. Following the lesson, an interview with the biology teacher was conducted. The interview followed a semi-structured interview protocol that included questions regarding learning goals, instructional strategies, students' understanding, and assessment and questions to capture the teacher's attitude towards the CT integrated biology lesson. The interviews were audio recorded and transcribed verbatim. The analysis procedure started with an evaluation of the correctness and efficiency of the decision trees to investigate students' learning outcomes regarding cell types. The decision trees, surveys, exit tickets, and students' interviews were used for the analysis of students' learning outcomes regarding CT and students' attitude towards the CT integrated biology lesson. The analysis aimed at capturing students' experiences and the variation in students' ideas. The interview data were analyzed inductively. When analyzing the teacher's attitude and views, four categories were discovered.

3. RESULTS

In this section, we first report the results regarding the effect of using decision trees on students' biology and CT learning outcomes. Furthermore, the results of the analysis of students' and teacher's views about the CT integrated lesson are reported. The students' quotes were translated from Dutch into English. This also applies to the translated text in the decision trees except for 'ja' (yes) and 'nee' (no) (Figures 2-5).

3.1. Biology Learning Outcomes

Students had drawn decision trees to classify cell types and these drawings were used to analyze students' biology learning outcomes. A correct decision tree with as few questions as possible should include three questions about cell nucleus, cell wall, and chloroplasts. 13 out of 18 students made a decision tree with three questions. Questions could be asked in different order, for example in Figure 2 the first question is about chloroplasts while in Figure 3 the first question is about the cell wall. Seven out of these 13 students did not use correct questions, for instance, they asked whether there was a vacuole or cytoplasm, which are no distinctive characteristics of cells.

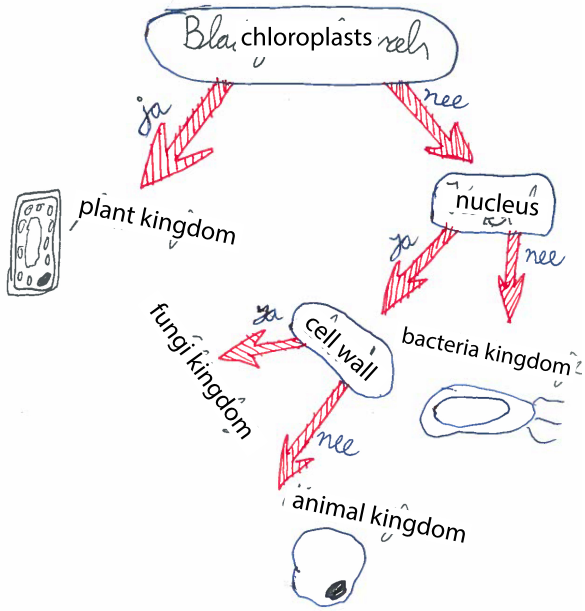


Figure 2. Example of correct decision tree (Student [S]16)

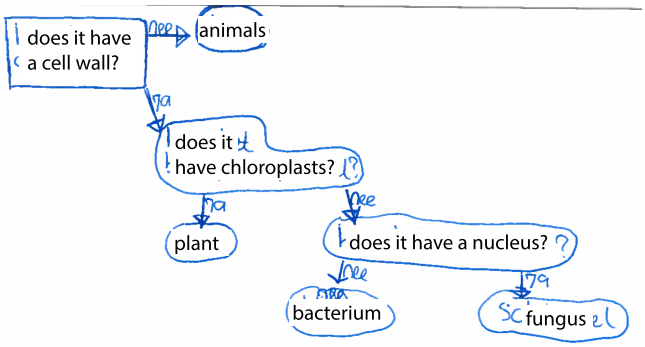


Figure 3. Example of correct decision tree with questions in different order (S10)

Furthermore, five out of 18 students constructed a decision tree with four or five questions. This reveals that it is not obvious for students to design suitable questions for determining cells. In addition, it shows that asking students to draw a decision tree may be a helpful instrument for formative assessment because misconceptions about distinctive characteristics are easily visible.

3.2. CT Learning Outcomes

The analysis of students' decision trees, completed surveys and exit tickets, and the interview transcripts revealed students' learning outcomes regarding CT, especially regarding algorithmic thinking, abstraction, and evaluation.

Algorithmic thinking. All students were able to design a simple conditional algorithm in the shape of a decision tree. Because the questions could be asked in different order, different algorithms could be designed. In the interviews with students, we explicitly asked whether they realized that multiple decision trees could be correct. But a question about differences between student's own decision tree and the decision tree of peers was interpreted by a student as difference in appearance: *"she used circles and I had drawn text and arrows"* (Student [S]15). Other students realized that several solutions are possible but did not really know how to explain why there are more options. When asked why then they would choose one or the other, one student replied: *"maybe because you're more used to it or it's easier"* (S2).

Abstraction. All students were able to use a decision tree to represent data at this abstract level. Another learning goal for abstraction was to separate important from redundant information. The decision tree could be made with three questions; however, some students did not omit unnecessary details. This is clear in the decision tree in Figure 4, where the last question is redundant.

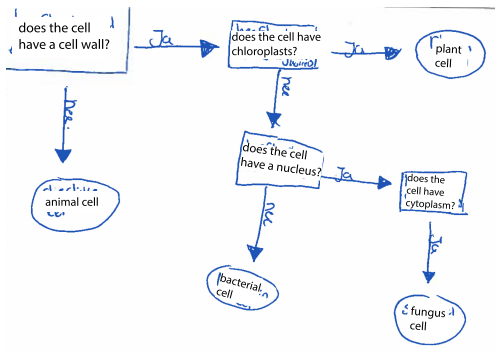


Figure 4. Decision tree with redundant question (S5)

Evaluation. Students were supposed to ask each other for feedback. Most of the given feedback is either a compliment ("*well and clearly made*") or is about the appearance of the drawing "*make it neater*". Some students then drew a new version. Figure 4 is such an improved version, the first version made by this student is shown in Figure 5. It is interesting to note that the questions are the same as in the first version (which means too many questions), but in Figure 4, rectangles are used for questions and ovals for the four kingdoms, which makes the structure/the algorithm much clearer.

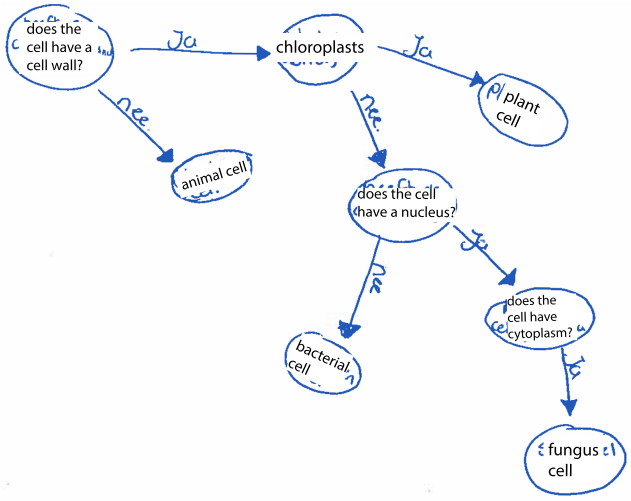


Figure 5. First version of decision tree (also made by S5)

3.3. Students' Attitude toward CT Integrated Biology Lesson

The analysis of the exit tickets completed by the students revealed that most students (16 out of 18) understood the lesson. 13 students reported that they understood the assignment. Ten students answered that the lesson was not difficult. Students were divided on the question whether they enjoyed the lesson (seven students enjoyed the lesson, seven students were not sure and four did not enjoy the lesson). More than half of the students (10 out of 18) found the lesson interesting, six students were not sure and two did not find it interesting.

The analysis of the questionnaires showed that most students were satisfied with their decision tree, either because "*it was clear*" (S6) or "*it was right*" (S13) and one student also described "*because I understand it better*" (S3). Students valued the way a decision tree helps with the determination of cells because *"it helps you get an overview"* (S16) or "*because you ask yourself questions and you answer them too*" (S10). During the interviews, one of the students commented that a decision tree helps because *"then you can look it up somewhere, because in your head you are already doing that"* (S15). And another student reported that a decision tree is helpful for the determination of cells because *"if you know the characteristics, it is easy to use; for example, when you know that there is no nucleus, then you can distract it* [the kingdom] *easily"* (S9).

In the questionnaire, students were asked if they could give examples where computers could use decision trees. A student answered that a decision tree might help for learning about animals (instead of cells) and another student replied

that it might be useful *"for finding the right information"* (S7).

3.4. Teacher's Attitude and Views toward CT Integrated Biology Lesson

Results of the teacher's attitude and views are presented according to the categories that emerged from the inductive analysis.

Learning goals: The teacher's main goal for the students was to learn about cells and their characteristics and to be able to distinguish the four kingdoms. According to the teacher, the integration of CT in the lesson was supportive of this biology learning. Last year, the students had used a decision tree or search map, but this time, students were asked to draw the decision tree themselves. Structuring the information was hard for students but it contributed to their learning about cell types.

Way of learning: The teacher described that by asking students to draw a decision tree, they learn a different way of thinking: "*they had to reason it in a certain way, rather than memorizing it or making a guess; and I think I addressed that* [the reasoning]". According to the teacher, this reasoning is especially relevant for learning biology because there are many topics that you can either memorize or reason logically, for instance regarding blood types. "*Biology is nothing but logical reasoning, if this… then that…then this… And this is a very nice way to deal with that, specifically for students of this level of education.*"

Assessment: The teacher noticed that asking the students to draw a decision tree gave him a better insight in the students' understanding. When he previously covered this lesson topic, he would use interactive instruction and only get an idea of the understanding of the three students that would participate in the conversation. This lesson, however, will provide him with an understanding of all students and "*much more of their way of thinking*". Therefore, the decision trees did not only support students in their learning, but also provided the teacher with an improved awareness of students' understanding.

CT integration: The teacher appeared to have a positive opinion of the integration of CT in the biology lesson: "*the integration, that's just very important to me*". He valued the integration because the learning of CT helped students learn biology. During the interview, the teacher also commented on the way CT was integrated in the biology lesson. In the lesson, students used pen and paper to draw a decision tree, which was fine according to the teacher because "*learning CT doesn't always have to be digital*".

4. CONCLUSION AND DISCUSSION

In this study, we aimed to explore students' learning outcomes about biology and CT, as well as students' and teacher's attitudes towards the CT integrated biology lesson. Regarding the biology learning outcomes, the results reveal that all students struggled to design good questions for classifying the different types of cells. Some students have difficulty to understand the distinctive characteristics of different cell types. Among the many different approaches to teach biology to secondary school students, decision trees seem a useful tool for teachers to help students to structure their knowledge instead of memorizing the knowledge. While creating decision trees by using the biology knowledge, students are exposed to high-level thinking activities (such as analysis, synthesis, evaluation) (Bloom, 1956). Additionally, in this lesson, the teacher encouraged students to create their own decision tree via less-structured scaffolding. However, depending on the teaching approach of the teacher, the complexity of the subject, or the development level of students, more guided/structured scaffolding can be offered during the design stage of the decision tree to avoid misconceptions and mistakes.

Regarding CT learning outcomes, the results show that students were able to design simple algorithms by using if-statements, which improves the algorithmic thinking skills of students. Some students made a clear visualization, which helps to improve abstraction skills of students. Some students could not separate redundant from important information and added some unnecessary questions in their decision trees which is related to their abstraction skill. Related to the visualization aspect of decision tree models, it is a well-known fact that the ability to effectively use visualizations is an important aspect of computational thinking, particularly as it relates to the STEM fields (NRC, 2011). In addition, the ability to create, refine, and use models of phenomena is a central practice for scientists and mathematicians (NGSS Lead States, 2013). The process of designing a model involves making methodological and conceptual decisions and there are many reasons that might motivate designing a model, including wanting to better understand a phenomenon, to test out a hypothesis, or to communicate an idea or principle to others in a dynamic, interactive way (Weintrop et al., 2016). The results also revealed that students' evaluation skills could be improved. Their evaluations and feedback were mostly related to the visual design and were much less related to biology content or algorithms.

Overall, the general attitude of students toward the CT integrated biology lesson was positive and they found it interesting. Students valued the way a decision tree helps with the determination of cells. Also, the teacher's views about the lesson are quite positive and he described that a decision tree helps to teach reasoning which is a very useful skill for biology lessons. It also offers a way of formative assessment and provides the teacher with an improved awareness of students' understanding. An unplugged decision tree is easy to use for teachers and students. The use of decision trees shows that is not necessary to have full programming or IT knowledge to be able to integrate CT into a disciplinary context.

5. REFERENCES

Augustine, N. R. (2005). Rising above the gathering storm: Energizing and employing America for a brighter economic future. Washington, DC: National Academy Press.

Barab, S., Thomas, M., Dodge, T., Carteaux, R., & Tuzun, H. (2005). Making learning fun: Quest Atlantis, a game without guns. *Educational technology research and development, 53(1),* 86-107.

Barendsen, E. (2022). Computational Thinking in context: a teaching and learning trajectory for primary and secondary education. Retrieved from https://www.nwo.nl/projecten/40518540153-0

Basu, S., Biswas, G., Kinnebrew, J., Rafi, T. (2015) Relations between modeling behavior and learning in a Computational Thinking based science learning environment. In *Proceedings of the 23rd International Conference on Computers in Education*, pp. 184–189.

Blikstein, P. (2013). Digital fabrication and 'making' in education: The democratization of invention. In: Walter-Herrmann J. & Büching C. (Eds.), *FabLabs: Of Machines, Makers and Inventors.* Bielefeld: Transcript Publishers.

Bloom, B. S. (1956). Taxonomy of educational objectives: The classification of educational goals. *Harlow, England: Longman Group*

Buechley, L., Eisenberg, M., Catchen, J., & Crockett, A. (2008). The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 423-432).

Che, D., Liu, Q., Rasheed, K., & Tao, X. (2011). Decision tree and ensemble learning algorithms with their applications in bioinformatics. In: Arabnia H., Tran QN. (eds) *Software tools and algorithms for biological systems*, 191-199.

Dudoit, S.J., & Fridlyand, J. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association, 97*:77–87.

Foster, I. (2006). A two-way street to science's future. *Nature*, *440*(7083), 419.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42(1),* 38–43.

Jacobson, M. J., Kim, B., Pathak, S., & Zhang, B. (2015). To guide or not to guide: issues in the sequencing of pedagogical structure in computational model-based learning. *Interactive Learning Environments, 23(6),* 715-730.

Kafai, Y. B., Lee, E., Searle, K., Fields, D., Kaplan, E., & Lui, D. (2014). A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles. *ACM Transactions on Computing Education (TOCE), 14(1),* 1-20.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2(1),* 32-37.

Lehmkuhl-Dakhwe, K. V. (2018). An instructional framework, model lessons, and professional learning program for science standards-aligned computing in 4th - 12th grade classrooms. In *2018 IEEE Frontiers in Education Conference (pp. 1-5)*.

NGSS Lead States (2013) Next generation science standards: for states, by states. The National Academies Press, Washington, DC

National Research Council (NRC) (2011) Report of a workshop of pedagogical aspects of computational thinking. The National Academies Press, Washington, DC

Othman, S.M., Ba-Alwi, F.M., Alsohybe, N.T., & Al-Hashida, A.Y. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. *Journal of Big Data*, *5:34*, 1-12.

Patton, M. (2002). Qualitative research and evaluation methods (3rd ed.). *Thousand Oaks, CA: Sage publications.*

Peel, A., Fulton, J., & Pontelli, E. (2015). DISSECT: An experiment in infusing computational thinking in a sixth grade classroom. In *2015 IEEE Frontiers in Education Conference (pp. 1-8).*

Petrović, J., & Pale, P. (2017). Decision trees in formative procedural knowledge assessment. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 17-20).

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, *52(11),* 60-67.

Salzberg, S., Delcher, A.L., Fasman, K.H. and Henderson, J. (1998) A decision tree system for finding genes in DNA, *Journal of Computational Biology*, 5, 667–680

Stake, R., E. (1994). Case study: Composition and performance. *Bulletin of the Council for Research in Music Education*, 31–44

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25(1),* 127-147.

Weintrop, D., & Wilensky, U. (2013). RoboBuilder: a computational thinking game. In *Proceedings of the 44th SIGSCE Technical Symposium on Computer Science Education (p. 736*).

Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., ... & Yaschenko, E. (2006) Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, *34(suppl_1)* D173–D180

Wilensky, U., & Rand, W. (2015). An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo. *MIT Press.*

DOI: 10.34641/ctestem.2022.461

A STEM-based Learning Activity Instructional Design of Quadruped Bionic Robots

Shaun-Wen Chen*, Ju-Ling SHIH
Graduate Institute of Network Learning Technology, National Central University, Taiwan
*seanchen54017@gmail.com, juling@cl.ncu.edu.tw

ABSTRACT
This study designs a STEM–based learning activities related to bionic robots and examines students' learning performance of the instructional design. With the rapid development of science and technology, robots play an important role in human society, helping people in solving repetitive work using automatic objects in biological mechanical structure. However, quadruped bionic robots are usually expensive due to its mobility and stability control of locomotion. In this study, a quadruped bionic robot is designed using the linkage mechanism using 3D printing combined with the Micro:bit control board for motion. The purpose of this study is not only to experiment the making of a bionic robot, but also to construct an instructional design for the first-time STEM and robotics learners to learn basic mechanical structure, fabrication process, and programming using MakeCode. This production process is expected to inspire the students' learning motivation in the robotic production, and to improve mechanical concept and computational thinking.

KEYWORDS
Bionic Robot, STEM, Linkage Mechanism, Instructional Design, Computational Thinking

1. INTRODUCTION
STEM education has received a lot of attention recently. In order to be able to deal effectively with problems in the STEM field, the importance of computational thinking (CT) skills is highlighted. STEM is also considered to be a factor that can affect technological and economic development (Xie, Fang, & Sauman, 2015). Robots are also a technological trend in today's society (Li et al., 2011). This study believes that the technology of bionic robots is particularly important, since bionic robots imitate the walking methods of animals in nature, so it can adapt to a variety of special terrains. But in order to reduce the cost of making and to release cognitive loads, the linkage mechanism will be taught to construct bionic robot legs.

In this activity, students will learn about STEM, computational thinking, bionic robots, linkage mechanisms and related algorithmic concepts. Students will also build their first bionic robot. During the activity, students will be provided with components to assemble robots with their own design, and they will be guided to find out how to make their bionic robots move faster and more stable across various terrains. This paper mainly uses qualitative analysis and supplemented by quantitative analysis to explore the following research questions:

1. What are the processes and strategies used by learners in the process of making bionic robots? (Qualitative analysis)
2. What is the level of understanding and absorption of the linkage mechanism by the learners? (Quantitative analysis)

2. LITERATUR REVIEW

2.1. Computational Thinking
Computational Thinking has become an important cognitive skill to develop relating to all areas of education. Computational thinking can be traced back from Wing (2006) by saying that computational thinking is to allow humans and computers to cooperate and solve problems instead of thinking like a computer. Nowadays "computational thinking affects research in almost every discipline, including the sciences and humanities" (Bundy, 2007, p.67). Bundy stated that computational thinking is an important skill for problem-solving and thinking skills that apply in multiple disciplines. To learn computational thinking skills, education is important to enhance and reinforce intellectual skill so that CT can be used in various disciplines (Wing, 2011). Selby and Woollard (2013) divided CT into five major themes: abstraction, decomposition, algorithm, evaluation, and generalization.

1. **Abstraction** is about creating and defining the relationships between problems and formulating rules that can be solved step-by-step for similar problems and implemented repeatedly to simplifying information (Council, 2010); and displaying only the information that is needed (Peel & Friedrichsen, 2017).
2. **Decomposition** is the classification of potential elements to determine the substantive elements and the relationship between elements. Different strategies are used to decompose such as means-end, bottom-up, multivariate, and etc. (Rich, Egan, & Ellsworth, 2019).
3. **Algorithm** is a sequence of steps to solve a problem (Peel & Friedrichsen, 2017), and to develop rules that can solve similar problems step by step in order to be implemented repeatedly.
4. **Evaluation** is the process of ensuring that algorithms and solutions are feasible. Various properties of algorithms need to be evaluated, including whether they are correct, fast enough, use resources efficiently, and easy to use.
5. **Generalization** is the process of creating models, rules, principles, or patterns of observation to test predictions; and the step to identify how some small pieces can be

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

repurposed and reapplied to similar or unique problems (Selby & Woollard, 2013).

"CT is an essential skill for navigating today's complex technological world (Peel & Friedrichsen, 2017, pp.21)." The purpose of this study is to encourage junior high school students to learn to use computational thinking. Through the above five skills to complete the activities held by this study, students can effectively use CT ability when encountering problems in various disciplines.

2.2. STEM-Based Instructional Design

STEM is composed of four core fields: Science, Technology, Engineering, and Mathematics (Xie, Fang, & Sauman, 2015). Increasing attention has been paid on STEM education in recent years. For students to deal with the problems encountered in real-life more effectively, STEM is a must-learn course (Julià & Antolí, 2019). The instructional design of STEM education has become an important part. How to design a course that can attract students to learn and can effectively strengthen students' ability in STEM has become an important topic for teachers. Khalil and Elkhider (2016) identified five pedagogical principles of instruction:

1. The learner is dedicated to engage in solving real-world problems.
2. Activate existing knowledge as a basis for new knowledge.
3. Demonstrate new knowledge to learners.
4. Learners applied the new knowledges.
5. Learners integrated the new-learned knowledge into the learner's world.

There are many types of instructional designs, among which project-based learning is a design that effectively helps learners learn from the process of creating a project (Guo et al., 2020). The part of making a project is the same as this study, so the project-based learning theory is used to design the activities of the study. The project-based learning approach to learning concentrates on constructing and explaining meaning, and that (1) knowledge is constructed; (2) it must be preceded by the teaching of prior knowledge; (3) the whole is slowly constructed from the parts; (4) requires effort to engage in purposeful activities to build useful knowledge structures (Gómez-del Río & Rodríguez, 2022). The activities in this study were designed based on the above STEM-based instructional design combined with project-based learning, and the activities allowed students to learn computational thinking skills.

2.3. Bionic Robot with Linkage Mechanism

In recent years, legged robots are getting more attention. Robots with wheels are fast and easy to control, but they cannot adapt to rough terrains. In contrast, legged robots can adapt to a variety of different terrains, and more and more people are developing diverse kinds of legged robots for various situations. For example, a bionic cheetah is developed by Massachusetts Institute of Technology (MIT) that can walk on rough terrain (Singh & Kotecha, 2020). Mammals including humans have developed a unique way of walking, which can effectively adapt to various terrains in nature (Li et al., 2011). Therefore, the bionic leg mechanism becomes the best reference for the design of walking robots. However, the high development cost and complex control limit the prevalence and application of this type of robot. Therefore, the linkage mechanism of the leg has become a new research direction. Ujjiban Kakati (2015) designed a two-leg walking chair using linkage mechanism that can replace a wheelchair. It can adapt to various terrains due to the national geographic conditions of India. Its price is the same as the basic price of existing wheelchairs in India providing convenience for the physically challenged people to improve their lives. The value and potential of using bionic robot legs with linkage mechanism was successfully demonstrated.

Based on the above examples, this research believes that it is particularly important to establish some basic robotics knowledge for modern students, especially the related knowledge of bionic robots and linkage mechanisms, and there are few studies on using bionic robots in instructional design. Therefore, this research develops a bionic robot to give students inspirations when learning related knowledge, and to guild students to make, improve, and elevate their self-designed bionic robot.

3. RESEARCH DESIGN

3.1. Research Process

This study will be conducted in a junior high school in Taiwan with about 20 students. Students will be divided into 3 to 4 groups, and each group has 6 students. The learning activity begins with a simple quiz with five questions about the course to define students' prior knowledge level. Then the whole course will proceed from 2 to 4 hours depends on the students' levels. After the students have basic knowledge, they will carry on with the programming activity of the bionic robot, which takes about 90 minutes. After experiencing the bionic robot test-drive, and learning from robot model, students will spend about 2.5 hours making their own bionic robot. Then, the teacher will conduct a competition between groups for about 30 minutes. Finally, after the competition, the teacher will guide the students to review the key points learned from this activity and reflect on their own production. Questionnaires will be distributed to receive students' feedbacks, and analyze the students' learning effectiveness through post-test (as shown in *Figure 1*).

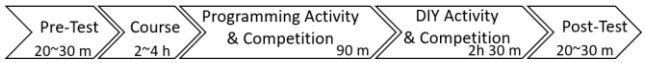


Figure 1. Activity Process

4. Activity DESIGN

4.1. Learning Objectives

Students are expected to learn CT skills as well as the knowledge of making bionic robots. The learning objectives include:

1. Understand what a four-bar linkage is, and how to build robot with the linkage mechanism;
2. Chose the best leg model to construct;
3. The ratio of motor hand for bionic robot;
4. The ratio of the linkage bars to move faster;

5. The key component of the leg mechanism that influences bionic robot's speed;
6. The process and strategies when students improve their bionic robot's speed.

These skills when constructing their bionic robots can be analyzed according to CT skills used in these activities:

1. Understand course content by finding out the relationship between each type of four-bar linkage. (Abstraction)
2. Break down the parts of a four-bar linkage to gain a better understanding of each part's function. And to figure out what will happen when different parts are connected in a different way. (Decomposition)
3. Students must make their own algorithm when coding the bionic robot, for example: first turn left, next go forward, then turn right, finally go forward. (Algorithm)
4. Revise the code after the first algorithm is tested. (Evaluation)
5. After students finish making their first bionic robot, they will have to test it out, then make some adjustment to improve it. (Evaluation)
6. Students will gain some knowledge on adjusting which part of the linkage to make it go faster after some try-and-errors. (Generalization)

After the activity is completed, the teacher collects the learning sheets which is also an assessment to students' CT performances. The pre-test and post-test results can show how well students learn from the activity.

4.2. Course Teaching for the Activities

After the establishment of prior knowledge of bionic robots, students participate the following activities. The topics, objectives, and contents of the course are shown in Table 1.

Table 1. Topics, objectives, and contents of the course

Topics	Contents	Objectives
STEM	The origin of STEM, definitions, meanings.	Knowing how this activity is related to various fields in STEM.
Bionic Robots	Basic concept and applications of bionic robots. Introduction of Bionic beasts and bionic robots. Bionic robot with four-bar linkage mechanism.	Understand the basic knowledge of bionic robots.
Linkage Mechanism	Introduction of Linkage Mechanism and how to apply it. Four-bar linkage mechanisms: Crank-Rocker mechanisms, Double-Crank mechanisms, Double-Rocker mechanisms.	Establish the knowledge of the linkage mechanism.
Motors	Introduction of DC Motors and Servo Motors.	Learn about the motors that will be used in the upcoming activities.
Micro:bit and Block Programming	Introduction of Micro:bit. MakeCode block programming.	Learn about the possibilities and basic programming of Micro:bit.

The five topics are described in details as below.

4.2.1. STEM

In this session, students learn about the origin, definitions, and the meanings of STEM, and understand how this activity is related to various fields in STEM. For example, when students make their own bionic robots, they are experiencing the "Engineering" part in STEM field.

4.2.2. Bionic Robots

In this session, students understand the basic concept of bionic robots, and the application of robots. Bionic beasts and bionic robots are introduced, among which the robots with four-bar linkage mechanism (Terefe, Lemu, & K/Mariam, 2019) are emphasized.

4.2.3. Linkage Mechanism

In this session, knowledge of linkage mechanism is introduced, including basic terms and four-bar linkage definitions, classifications, and variations. Professional terms, such as crank and rocker, are mentioned and demonstrated (as shown in *Figure 2*) so the students have sufficient knowledge to make their first bionic walking robot in the following DIY activities.

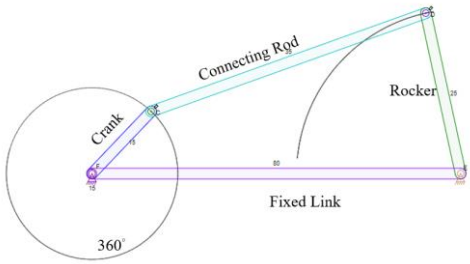


Figure 2. Basic Four-Bar Linkage Terms

There are types of four-bar linkage mechanisms (Martinez et al., 2012), for example: crank-rocker mechanism, double-crank mechanism, and double-rocker mechanism (as shown in *Figure 3*). They have common features, when the sum of the longest rod and the shortest rod is less than or equal to the sum of the other two rods, then it can form a four-bar linkage. When the fixed link is the dual rod of the shortest rod (the neighbor of the shortest rod), it is crank-rocker mechanism. When the fixed link is the shortest rod, it is a double-crank mechanism. When the fixed link is on the opposite side of the shortest rod, it is double-rocker mechanism.

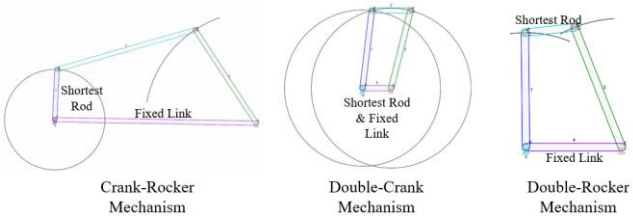


Figure 3. Three Types of Four-Bar Linkage Mechanism

In order to give students inspirations in making their own bionic robots, this session introduces two types of four-bar linkage leg designs: M-shaped four-bar linkage and Cross-shaped four-bar linkage. The M-shaped linkage is a double-rocker mechanism, and the Cross-shaped linkage is a crank-rocker mechanism (as shown in *Figure 4*).

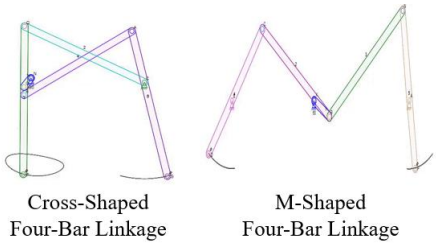


Figure 4. Cross-Shaped & M-Shaped Four-Bar Linkage

4.2.4. Motors

Motors are the basic equipment of most robots (Wahde, 2012). This session introduces basic motor knowledge to students focusing on the various types of motors. This activity introduces two types of motors, DC motors and servo motors. During the bionic robot programming activity, students will use Micro:bit development software MakeCode to write block programming to control servo motors; and use DC motors to mobilize their first bionic robot in the DIY activity.

4.2.5. Micro:bit and Block Programming

In order to control the motor, a control board is required. This activity uses Micro:bit to connect to the Micro:bit expansion board, Mbitbot. The pins of the servo motor are directly connected to Micro:bit. Students are taught to use MakeCode to write block programming (as shown in *Figure 5*) to control the motor movements.

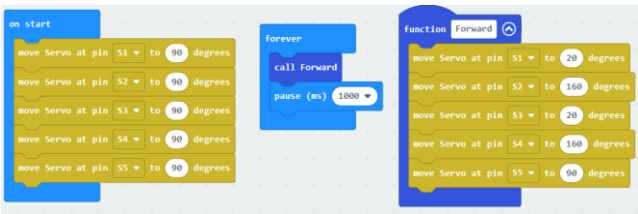


Figure 5. MakeCode Block Programming Example

4.3. Bionic Robot Programming Activity

The experience of using a program to control the movement of a bionic robot may give students inspirations when making their own bionic robots. In order to allow students to experience how a basic bionic robot movement and to see how a professional structure is constructed, a bionic robot with eight-bar linkage was developed in this study.

This bionic robot is mainly developed with reference to " Eight-Bar Linkage Walking Mechanism (Yan, 2007)". First, sketch the robot components on SolidWorks before using 3D-printing to produce the parts. Then, assemble the parts with motors, Micro:bit and Mbitbot. Finally, write programming to mobilize the bionic robot. The constructing process is shown as Figure 6 below.

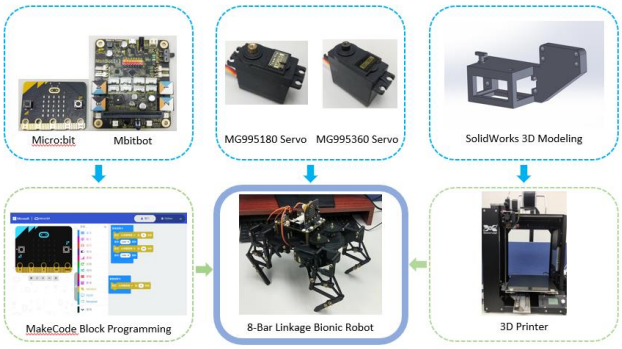


Figure 6. Bionic Robot Constructing Process

The bionic robot programming activity is for students to write the block program in MakeCode to control the bionic robot through obstacles. Students will be divided into groups of three, controlling the bionic robot to move from the starting point to the ending point, and return to the starting point after obtaining a component card. Students need to get three component cards including: linkage rod set 1, linkage rod set 2, and motors. The group that gets the three component cards and returns the fastest wins. The winning team can gain advantages in the competition session of the DIY bionic robot.

4.4. Bionic Robot DIY Activity

In this session, students use the component cards they obtained to exchange with the teacher for the components of bionic robots including linkage rod sets and motors. Students work on their own and use these parts to assemble their first bionic robot. Students can make the M-shaped four-bar linkage or Cross-shaped four-bar linkage taught in the course, or they can use their own creativity to make other linkage mechanism. During the process, a learning sheet will be distributed to the students which not only guide them through the production process but to document the changes of their designs and difficulties that they encounter as well as explain why and how they make adjustments. Reflection questions are also included to help students reflect on their designs, such as: What if I extend the length of the crank, will the bionic robot walk faster? During the construction process, the teacher will give scaffolding assistance (Sawyer, 2005) so that students can maintain interest and curiosity in the learning process.

After the students have assembled their own bionic robots, they will participate an in-class competition. Students will place their own bionic robots at the starting line. After the robot reaches the finish line, the students will answer some questions about the course. If the students answer correctly, they can turn the robot around and go back to starting line to get the points. If they answer incorrectly, the teacher will teach the students about the correct answer. Each successful round counts as one point, and the student who earns the most points within five minutes or who answers all the questions the fastest wins. The winner will advance to reach the final championship and get a small prize from the teacher.

5. EXPECTED RESULTS

5.1. CT Performances

In order to evaluate how students' CT performance are, students will have to document the constructing process. With the documents students wrote, teachers can find out

how well individuals perform. This research expects that students can gain better understanding of the course content through the learning worksheet and guides (Learn to do "Abstraction" and "Decomposition" in CT skills.). They can learn how the linkage moves and the bionic robot's architecture in the programming activity (Learn to do "Abstraction", "Decomposition", and "Algorithm" in CT skills.). When students are making their own bionic robots, they can improve their linkage concept through continuous experiments and evaluations (Learn to do all five CT skills.).

5.2. Knowledge Acquisition of Bionic Robot with Linkage Mechanism

There will be a pre-test and post-test before and after the learning activity. The difference between the pre-test and post-test can be used to observe whether the students have acquired the relevant knowledge of the bionic robot with linkage mechanism after this activity. Questionnaire include basic linkage mechanism and the understanding of bionic robots, for example: What is a four-bar linkage mechanism? This research expects students to acquire basic linkage knowledge and comprehend the knowledge of bionic robots through a series of activities, and gain inspiration for their future projects or designs.

5.3. Overall Satisfactions

At the end of the activity, students will be asked to fill out a questionnaire. The questions include their thoughts and general feelings about each stage of the activity. Students will also be asked if there is any room for improvement in each stage of the activity, and it is expected that the content of this activity can be adjusted through the feedback of the students.

6. CONCLUSION

Nowadays, technology is developing rapidly, modern people will have to learn interdisciplinary skills. Students need to develop basic exploration, research, and problem-solving skills, of the 21st century information literacy. In order to keep up with the progress of the times, this research will guide students to explore the field of bionic robots. For example, the impact on bionic robots when changing the proportions and length of various components, and give scaffolding (Sawyer, 2014) to assist students in learning.

Technology products can be seen everywhere in life. No matter one is an expert or an ordinary worker, learning CT skills and understanding STEM fields can make one be more competitive. Computational thinking skills can allow one to effectively solve problems in life. Therefore, it is hoped that students can develop relevant abilities through this activity.

In this learning activity, learners will explore the field of STEM and CT from scratch by designing and creating their own bionic robots. This study expects that learners can increase the ability of computational thinking and master the basic knowledge of linkage mechanism, and enhance learning motivation. This study expected to provide a curriculum model as a reference for future instructional designs, especially of STEM-related activities, so that young learners can be proficient problem-solver.

7. REFERENCES

Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing, 1*(2), 67-69. http://www.inf.ed.ac.uk/publications/online/1245.pdf

Council, N. R. (2010). *Report of a Workshop on the Scope and Nature of Computational Thinking*. The National Academies Press. doi:10.17226/12840

Gómez-del Río, T., & Rodríguez, J. (2022). Design and assessment of a project-based learning in a laboratory for integrating knowledge and improving engineering design skills. *Education for Chemical Engineers*. https://doi.org/10.1016/j.ece.2022.04.002

Guo, P., Saab, N., Post, L. S., & Admiraal, W. (2020). A review of project-based learning in higher education: Student outcomes and measures. *International Journal of Educational Research, 102*, 101586. https://doi.org/10.1016/j.ijer.2020.101586

Gustafson, K., Branch, R. (2002). Survey of Instructional Development Models (4th ed.). *New York: Eric Publications*.

Julià, C., & Antolí, J. Ò. (2019). Impact of implementing a long-term STEM-based active learning course on students' motivation. *International Journal of Technology and Design Education 29*, 303–327. doi:10.1007/s10798-018-9441-8

Khalil, M. K., & Elkhider, I. A. (2016). Applying learning theories and instructional design models for effective instruction. *Advances in physiology education, 40*(2), 147-156. doi:10.1152/advan.00138.2015

Li, Y., Li, B., Ruan, J., & Rong, X. (2011, September). *Research of mammal bionic quadruped robots: A review.* In 2011 IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM), 166-171. doi: 10.1109/RAMECH.2011.6070476.

Martinez, E., Romero, C., Carbonell, M. V., & Florez, M. (2012, July). *On the geometry and design of four bar linkage mechanisms.* In Proceedings of the 4th International Conference on Education and New Learning Technologies Barcelona, Barcelona, Spain, 2-4.

Peel, A., & Friedrichsen, P. (2017). Algorithms, Abstractions, and Iterations: Teaching Computational Thinking Using Protein Synthesis Translation. *The American Biology Teacher, 80*(1), 21–28. doi:10.1525/abt.2018.80.1.21

Rich, P. J., Egan, G., & Ellsworth, J. (2019, July). *A framework for decomposition in computational thinking.* In Proceedings of the 2019 ACM conference on innovation and technology in computer science education, 416-421. doi:10.1145/3304221.3319793

Sawyer, R. K. (2014). *The future of learning: Grounding educational innovation in the learning sciences.* The Cambridge handbook of the learning sciences, 726-746.

Selby, C., & Woollard, J. (2013). *Computational thinking:*

the developing definition. Conference: Special Interest Group on Computer Science Education (SIGCSE), 2014. URL http://eprints.soton.ac.uk/id/eprint/356481

Singh, Rakesh and Kotecha, Radhika, Quadruped Robot Gait Planning for Enhanced Locomotion and Stability (April 8, 2020). Proceedings of the 3rd International Conference on Advances in Science & Technology (ICAST). doi:10.2139/ssrn.3572191

Terefe, T. O., Lemu, H. G., & K/Mariam, A. (2019). Review and synthesis of a walking machine (Robot) leg mechanism. *MATEC Web of Conferences, 290*, 08012. doi:10.1051/matecconf/201929008012

Wahde, M. (2012). *Introduction to autonomous robots.* Lecture Notes from the course Autonomous Agents, Chalmers university of technology.

Wing, J. M. (2006). Computational thinking. *Communications of the Acm, 49*(3), 33-35. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical Physical and Engineering Sciences, 366*(1881), 3717-3725. https://doi.org/10.1098/rsta.2008.0118

Wing, J. M. (2011). Research notebook: Computational thinking—What and why. *The link magazine, 6*, 20-23. https://people.cs.vt.edu/~kafura/CS6604/Papers/CT-What-And-Why.pdf

Xie, Y., Fang, M., & Shauman, K. (2015). STEM Education. *Annu Rev Sociol, 41*, 331-357. doi:10.1146/annurev-soc-071312-145659

Yan, H. S. (2007). Walking Machines. *Reconstruction Designs of Lost Ancient Chinese Machinery*, 269-302. doi:10.1007/978-1-4020-6460-9_8

DOI: 10.34641/ctestem.2022.462

Comparison of STEM, non-STEM, and Mixed-Disciplines Pre-service Teachers' Early Conceptions about Computational Thinking

Wendy HUANG*, Chee-Kit LOOI, Ibrahim H. YETER
National Institute of Education, Nanyang Technological University, Singapore
wendy.huang@nie.edu.sg, cheekit.looi@nie.edu.sg, ibrahim.yeter@nie.edu.sg

ABSTRACT

This paper presents the results of an investigation on pre-service teachers' conceptions of computational thinking (CT) in Singapore prior to a two-hour introductory module on CT. Of 407 teachers, 280 provided valid responses to the pre-survey, which included questions on teachers' school subjects, current understandings of CT, confidence in their understandings of CT, and sources of the understandings. We deductively coded the open-ended responses through thematic analysis using four categories from a synthesis review on teachers' preconceptions of CT. The participants were classified into three groups, including STEM (primarily sciences and mathematics), non-STEM (e.g., humanities and languages), and mixed-disciplines (e.g., science and English language arts). The findings of the pre-survey showed that 42% of respondents (n=118) reported no prior knowledge of CT. Among the remaining 162 responses, the most popular view of CT was problem solving using various kinds of thinking, such as "logic", "abstraction", "step-by-step", and "decomposition" (n=106). STEM and mixed disciplines teachers (33%) reported higher levels of confidence compared to non-STEM teachers (15%). A higher percentage of STEM (64%) and mixed-disciplines (60%) pre-service teachers indicated learning about CT from formal courses during their university studies or teacher training, compared to non-STEM teachers (52%). This suggests that schools of education can play a bigger role in expanding CT awareness among pre-service teachers from non-STEM backgrounds. Finally, implications for teacher education are widely discussed.

KEYWORDS

computational thinking, teachers, conceptions, STEM, survey

1. INTRODUCTION

In 2020, Singapore updated its plan on developing digital literacy in general education to include computational thinking (CT) (Learn for Life, 2020). Existing programs that teach computational thinking (CT) through coding, robotics, and physical computing were scaled to more schools. The Ministry of Education (MOE) produced a guide on teaching CT in secondary level mathematics, reflecting a popular approach of integrating CT into existing subjects (Huang et al., 2021; Lee & Malyn-Smith, 2020; Pollock et al., 2019; Sherwood et al., 2021), rather than as a standalone subject, or only in computing classes. As applications of computing has led to fundamentally new advances in knowledge production across disciplines (e.g., Arnold, 2020; Qin, 2020), integrating CT could provide new perspectives on various subjects of study, as well as prepare students with relevant work skills that could spur technological innovations across sectors.

To achieve these educational objectives, training and supporting teachers are essential activities. In 2017 and 2018, we developed a day-long module to introduce all graduating Singapore pre-service teachers to CT, through activities that included Scratch programming, unplugged games, and microprocessor programming. As the staffing requirement became unsustainable, we redesigned the module as a 3-hour interactive lecture that could be delivered by 2 instructors for cohorts of several hundreds.

To evaluate and guide the improvement of the module, a survey was administered prior to and immediately following each session. Since all pre-service teachers across subjects and levels were required to participate in the module, the survey responses could provide insights on early conceptions of CT held by different groups of teachers. The results of an earlier study showed differences in views of CT by STEM (science, technology, engineering, mathematics) teachers compared to non-STEM teachers (Looi et al., 2020). However, the study did not account for teachers who were trained in both a STEM and a non-STEM subject. Also, the open coding surfaced 65 labels, the majority of which had a frequency of 1 and were not included in the final analysis. For this new study, we included a third category of teachers ("mixed-disciplines") and accounted for all responses in the analysis. We investigated the following questions:

1. What are the differences in conceptions of CT between STEM, non-STEM, and mixed-disciplines pre-service teachers?
2. What are the relationships between confidence, source, and content of CT knowledge expressed by pre-service teachers?

2. BACKGROUND

As CT gained prominence as an educational objective for all students, more attention has been given to preparing educators to teach CT (e.g., Hestness et al., 2018; Yadav & Berthelsen, 2021). Barr and Stephenson (2011) proposed an expansive agenda to embed CT in the K-12 curriculum, calling upon the cooperation of multiple stakeholders. As a result, it is increasingly likely that teachers have *heard of* CT before *learning about* CT in a professional context. Also, teachers may have preconceptions based on the two words in the term itself. A better understanding of these early conceptions may help teacher educators and researchers anticipate and address them in professional learning or teacher preparation courses.

Cabrera (2019) synthesized recent literature on teachers' preconceptions of CT (e.g., Corradini et al., 2017; Garvin et al., 2019; Bower & Falkner, 2015; Yadav et al., 2014).

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

We adapted 4 relevant preconceptions, which were:

1. CT as technology integration
2. CT as equal to CS and programming
3. CT as a non-specific problem-solving strategy
4. CT as "thinking like a computer"

Of the 24 papers he reviewed, only one study contained all four preconceptions (Corradini et al., 2017). There were three other studies that had three of the four preconceptions (Bower et al., 2017; Garvin et al., 2019; Yadav et al, 2018). By adapting his categories to code our data, we could corroborate our findings with prior studies and demonstrate the usefulness of the categorization.

There was only one other study that we are aware of that compares the CT conceptions of STEM and non-STEM teachers (Sands et al., 2018). The researchers pre-identified ten conceptions and asked teachers how much they agreed that each counted as CT. They concluded that there was no difference between the two groups. In our study, we did not set out to determine how well teachers could identify correct and incorrect conceptions of CT. We asked the study participants to tell us their understanding of CT without imposing any constraints.

3. METHOD

3.1. Respondents

The National Institute of Education (NIE) is the sole teacher training institute in Singapore. In November 2021, of the 407 graduating pre-service teachers who attended the required Introduction to CT session, 280 provided valid responses to the pre-survey. Most of the teachers who participated in the study were trained to teach non-STEM subjects (n=164, 59%), which included English Language, Literature, General Paper, History, Social Studies, Geography, Economics, Mother Tongue (Mandarin, Tamil, Malay), Character and Citizenship Education, Art, Music, Drama, and Accounting. Teachers categorized as STEM teachers taught Science, Mathematics, and Computer Applications (n=45, 16%). The mixed-disciplines teachers were trained to teach an English subject and a Science or Mathematics subject (n=71, 25%) (see Figure 1).

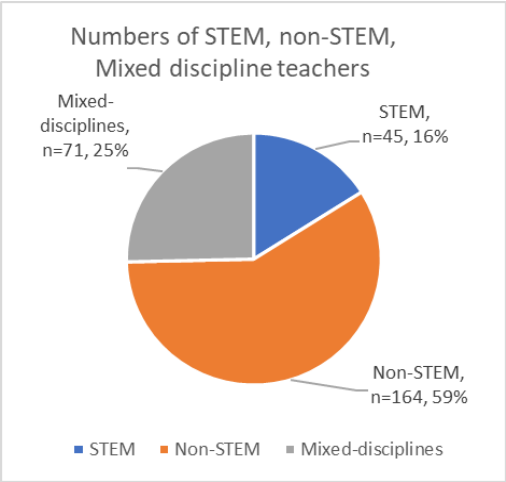


Figure 1. Numbers of teachers in each category

3.2. Survey

The survey consisted of four questions as shown in Table 1.

Table 1. Survey Questions.

Question	Response Mode
1. What subject areas have you been prepared to teach?	Check boxes
2. Rate your level of knowledge about "computational thinking"	Select radio button from 1 ("none") to 5 ("confident I can explain it")
3. What is your current understanding of "computational thinking"?	Open-ended text box
4. Where did you hear about "computational thinking"?	Open-ended text box

The respondents answered the questions using a google form. The data was cleaned and analyzed using spreadsheet software. Incomplete and vague responses were removed (e.g., "a little", "3"). For question four, 21 responses were considered invalid because the respondents reported no knowledge of CT but listed a source for hearing about CT.

Each response was assigned to one of three categories of teachers (STEM, non-STEM, mixed-disciplines). Each column could be filtered and sorted to explore potential relationships among the data.

We developed our codebook (Table 2) using the four categories of preconceptions (Cabrera, 2019).

Table 2. Codebook of CT Conceptions.

Code	Definition	Examples
Technology integration	Working with technological tools or studying technology. Using computer devices or software.	- "Application of computer software and data to work" - "Uses a computer to solve problems"
Computer Science or programming	Programming as the operationalization of CT. The thinking process of programmers. Thinking like a computer scientist. Applying CS techniques or principles when solving problems.	- "Knowledge about computer science?" - "Solving problems like a computer scientist, in a way that computers could also execute"
Problem solving or general thinking	Problem solving that involves higher order thinking skills such as abstraction, logical thinking, critical thinking, decomposition, among others. CT as a kind of problem-solving strategy that	- "How to solve problems systematically" - "Breaking problems down into simpler problems parsed in step-by-step terms a computer could

	enables people to create solutions for computational agents to carry out.	solve. Requires abstraction, i.e., the elimination of irrelevant details."
"Thinking like a computer"	Adopting the same process that a computer uses to "think". Understanding how a computer processes information so that humans can design instructions for computers to follow.	- "Thinking logically like a computer to complete problem-solving tasks" - "Breakdown of the thinking process that can be emulated by computers"
Other	Anything that doesn't fall in the above categories.	"Understanding how to use or apply formulas and (mental) operations in order to solve problems"

A sample (n=50) was independently coded by the three authors. Each response could be coded in more than category. 22 of the 50 responses had "none" for understanding of CT so were easily agreed upon. After resolving most of the differences among the remaining 28, the first author coded the remaining 230 responses.

The open-ended responses for question four were coded using constant comparison over several iterations, beginning with labels such as, "online", "TED ed", "Youtube", "reddit", which were then combined to form the category "internet / media". This process resulted in five categories, of which the other four consisted of "university", "NIE", "friends/family", and "guess".

4. FINDINGS

Of the 280 respondents, almost half (n=118, 42%) reported no prior knowledge of CT. Of these 118 "none" responses, non-STEM teachers made up a disproportionate amount (n=83, 70%), despite only representing 59% (n=164) of the total participants. Mixed-disciplines teachers were underrepresented, constituting 25% of the total participants (n=71) but only 20% (n=23) of the "none" respondents. STEM teachers were also underrepresented, constituting 16% of the total participants (n=45) while making up 10% of the "none" respondents (n=12). The results are shown Figure 2.

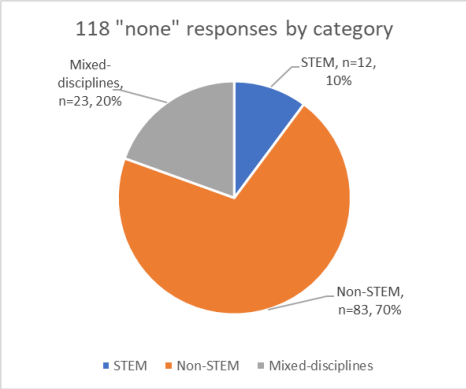


Figure 2. Distribution of teachers by category who reported having no knowledge of CT

Among those who reported having some conception of CT, there were gaps in confidence of their knowledge among the three groups of teachers (Figure 3 of the appendix). The results showed that overall, STEM teachers reported higher levels of confidence relative to mixed-disciplines and non-STEM teachers. 39% of STEM teachers (n=13) reported confidence levels of 4 and 5, compared to 27% of mixed-discipline teachers (n=13) and 15% of non-STEM teachers (n=12). In fact, zero non-STEM teachers reported a 5 in confidence. Additionally, 44% of non-STEM teachers (n=36) reported having the lowest possible level of confidence (level 2) in their conception of CT.

The results for CT conceptions are summarized in Figure 4, found in the appendix. The most popular conception was "problem-solving/thinking" (64% of coded responses) but there was variation among the three groups. For STEM teachers, this conception made up 67% of responses; for mixed-discipline, 73%; and non-STEM, 57%. Responses categorized in "technology integration" made up the smallest percentages (STEM: 3%; mixed-disciplines: 4%; non-STEM: 6%). Overall, the responses of non-STEM teachers were more varied across the four categories.

The results for sources of knowledge about CT are reported in Figure 5 of the appendix. The most popular source of knowledge about CT came from the teachers' university studies prior to the teacher training programme (37% of 148 valid responses). The STEM and mixed-disciplines teachers who named "university" had similar percentages (42% and 44%, respectively) compared to 31% of non-STEM teachers. These numbers may reflect recent university policies requiring undergraduate students to undertake coursework that includes CT. But there appears to be a gap between university students who pursued STEM versus non-STEM majors with respect to learning about CT. When combined with the percentages of responses that named NIE and other courses, 64% of STEM teachers learned about CT from a "formal" educational context compared to 60% of mixed-discipline teachers and 52% of non-STEM teachers. Also, 29% of non-STEM teachers heard about CT through an internet search or popular media, compared with 19% of mixed-disciplines teachers and 17% of STEM teachers. These statistics suggest a need for more non-STEM teachers to learn CT in a substantive context instead of through an internet search or popular media.

We observed an interesting relationship between confidence level and sources of CT knowledge. Teachers with the lowest confidence reported getting their knowledge of CT from friends/family by a factor of two relative to the responses of all teachers. However, the teachers who reported confidence levels of 4 and 5 relied one-third less on media/internet and 1.5 times more on their university studies relative to the responses of all teachers. Teachers who named NIE as their source of knowledge were half of those who reported "university" and less than those who learned about CT from the media/internet.

5. DISCUSSION

5.1. Addition of a "mixed-disciplines" category

Having a separate "mixed-disciplines" category did not seem to contribute much to this analysis. If we had combined the STEM and mixed-disciplines groups as a category of teachers trained to teach at least one STEM subject, the combined numbers would still constitute only two-thirds of the cohort but provide a better comparison with non-STEM teachers. For instance, the combined group would still be underrepresented among those reporting no knowledge of CT (i.e., 30% of "none" responses, but 41% of overall respondents).

5.2. Conceptions of CT

The four categories of CT conceptions fit our data well. Only 11 of the 280 responses were coded in the "other" category. Our findings showed that CT as "problem solving" was dominant (n=106), followed by much smaller numbers, "CS/programming" (n=23), "thinking like a computer" (n=20), and "technology integration" (n=8). As a cohort representing different subject areas and grade levels, our teachers' views were consistent with those reported by other researchers. Table 3 shows the top three or four preconceptions identified by a sample of similar studies, prior to professional development.

Table 3. Comparison with similar studies

Authors	Context	Top CT Preconceptions
(Yadav et al., 2014)	Pre-service teachers (control group, n=153)	Control group responses: "problem solving, logic", "use of technology, computers", "algorithms, step-by-step, directions"
(Yadav et al., 2017)	Pre-service teachers (n=134)	"problem solving", "logical thinking", "other types of thinking"
(Bower & Falkner, 2015)	In-service primary school teachers (n=32)	"problem solving with or using technology", "various types of thinking" (e.g., logical, analytical, mathematical)
(Bower et al., 2017)	In-service teachers (n=69)	"problem solving", "logical thinking", "coding", "using technology"
(Corradini et al., 2017)	In-service teachers (n=779)	"problem solving", "mental processes", "logical thinking", "algorithmic thinking"

Across groups, problem solving was the most common. We noticed that respondents often related mental processes, such as logical thinking, decomposition, algorithmic thinking, abstraction, and other forms of thinking to problem solving. However, our study had few teachers who thought of CT as "using technology or computers" compared to the teachers in other studies (e.g., Bower & Falkner, 2015; Bower et al., 2017; Yadav et al., 2014).

In our analysis, we could not conclusively claim that there were any significant differences between the conceptions of STEM, mixed-discipline, and non-STEM pre-service teachers. If we combined the teachers who were prepared to teach at least one STEM subject, 70% of them considered CT as problem solving compared to 59% of non-STEM teachers. The conceptions by non-STEM teachers were more varied across the four conceptions. There was a gap in the confidence levels between STEM and non-STEM teachers, which may be attributed to STEM teachers having more exposure to CT in formal education while non-STEM teachers relied more on information from the internet or friends.

We concur with Cabrera (2019) that what matters is not whether teachers have correct or incorrect ideas, but that their preconceptions are starting points for developing better understandings. The survey results on teachers' early conceptions can inform efforts to integrate CT into the pre-service teachers' curriculum studies. For instance, since problem solving was the most popular CT concept, we could help teachers identify which problems are suitable for applying CT in different subject areas. We could help teachers better understand the difference between how humans think and how computers process information. We could show how some uses of technology promotes CT. We could help teachers maintain a link between CT and computer science or programming without equating them.

6. LIMITATIONS

By giving respondents the option to say that they had no prior knowledge of CT, we likely missed out on other preconceptions that could have resulted in a different distribution among the four categories. The open-ended responses were sometimes difficult to interpret without more details from the teachers, such as interviews with a sample of the respondents from different categories. Our survey design lacked rigorous psychometric properties needed to uncover statistically significant differences between the groups of teachers. Hence, the findings' primary purpose is descriptive.

7. CONCLUSION

Our investigation corroborated existing studies of teachers' conceptions of CT prior to professional learning and contributed insights on relationships between preconceptions, knowledge source, and confidence levels. Many pre-service teachers were still not familiar with the term. Among those with some exposure, the greatest association of CT was with problem solving and various forms of thinking, followed by CS/programming, "thinking like a computer", and technology integration. Non-STEM teachers disproportionally made up more of the group who reported no knowledge of CT. A greater percentage of non-

STEM teachers also reported lower confidence levels than teachers who were prepared to teach at least one STEM subject. We attribute the gap to unequal access to formal learning about CT prior to or outside the teacher training programme. We argue that schools of education can therefore play a greater role in providing teachers with opportunities to learn CT as part of their curriculum studies. Rather than seeking to replace teachers' "misconceptions" about CT, teacher educators can design content that support teachers developing more nuanced and specific understandings.

8. REFERENCES

Arnold, C. (2020). How computational immunology changed the face of COVID-19 vaccine development. *Nature Medicine*. https://doi.org/10.1038/d41591-020-00027-9

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12. *ACM Inroads*, *2*(1), 48–54.

Bower, M., & Falkner, K. (2015). Computational thinking, the notional machine, pre-service teachers, and research opportunities. *Proceedings of the 17th Australasian Computing Education Conference (ACE 2015)*, 37–46.

Bower, M., Wood, L., Lai, J., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). *Improving the computational thinking pedagogical capabilities of school teachers*. *42*(3), 53–72.

Cabrera, L. (2019). Teacher Preconceptions of Computational Thinking: A Systematic Literature Review. *Journal of Technology and Teacher Education*, *27*(3), 1059–7069.

Corradini, I., Lodi, M., & Nardelli, E. (2017). Conceptions and Misconceptions about Computational Thinking among Italian Primary School Teachers. *Proceedings of the 2017 ACM Conference on International Computing Education Research*, 136–144.

Garvin, M., Killen, H., Plane, J., & Weintrop, D. (2019). Primary School Teachers' Conceptions of Computational Thinking. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 899–905.

Hestness, E., Ketelhut, D. J., Randy McGinnis, J., & Plane, J. (2018). Professional Knowledge Building within an Elementary Teacher Professional Development Experience on Computational Thinking in Science Education. *Journal of Technology and Teacher Education*, *26*(3), 411–435.

Huang, W., Chan, S. W., & Looi, C. K. (2021). Frame Shifting as a Challenge to Integrating Computational Thinking in Secondary Mathematics Education. *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, 390–396.

Learn for Life – Ready for the Future: Refreshing Our Curriculum and Skillsfuture for Educators. (n.d.). Base. Retrieved March 14, 2022, from https://www.moe.gov.sg/news/press-releases/20200304-learn-for-life-ready-for-the-future-refreshing-our-curriculum-and-skillsfuture-for-educators

Lee, I., & Malyn-Smith, J. (2020). Computational Thinking Integration Patterns Along the Framework Defining Computational Thinking from a Disciplinary Perspective. *Journal of Science Education and Technology*, *29*(1), 9–18.

Looi, C.-K., Chan, S. W., Huang, W., Seow, P., & Longkai, W. U. (2020). Preservice Teachers' Views of Computational Thinking: STEM Teachers vs non-STEM Teachers. In S. C. Kong, H. U. Hoppe, T. C. Hsu, R. H. Huang, B. C. Kuo, K. Y. Li, C. K. Looi, M. Milrad, J. L. Shih, K. F. Sin, K. S. Song, M. Specht, F. Sullivan, & J. Vahrenhold (Eds.), *Proceedings of International Conference on Computational Thinking Education 2020* (pp. 73–76). The Education University of Hong Kong.

Pollock, L., Mouza, C., Guidry, K. R., & Pusecker, K. (2019). Infusing Computational Thinking Across Disciplines: Reflections & Lessons Learned. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 435–441.

Qin, H. (2020). Machine learning and serving of discrete field theories. *Scientific Reports*, *10*(1), 19329.

Sands, P., Yadav, A., & Good, J. (2018). Computational Thinking in K-12: In-service Teacher Perceptions of Computational Thinking. In M. S. Khine (Ed.), *Computational Thinking in the STEM Disciplines: Foundations and Research Highlights* (pp. 151–164). Springer International Publishing.

Sherwood, H., Yan, W., Liu, R., Martin, W., Adair, A., Fancsali, C., Rivera-Cash, E., Pierce, M., & Israel, M. (2021). Diverse Approaches to School-wide Computational Thinking Integration at the Elementary Grades: A Cross-case Analysis. *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, 253–259.

Yadav, A., & Berthelsen, U. D. (2021). *Computational Thinking in Education: A Pedagogical Perspective*. Routledge.

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational Thinking in Teacher Education. In P. J. Rich & C. B. Hodges (Eds.), *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 205–220). Springer International Publishing.

Yadav, A., Krist, C., Good, J., & Caeli, E. N. (2018). Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, *28*(4), 371–400.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational Thinking in Elementary and Secondary Teacher Education. *ACM Trans. Comput. Educ.*, *14*(1), 1–16.

9. APPENDIX

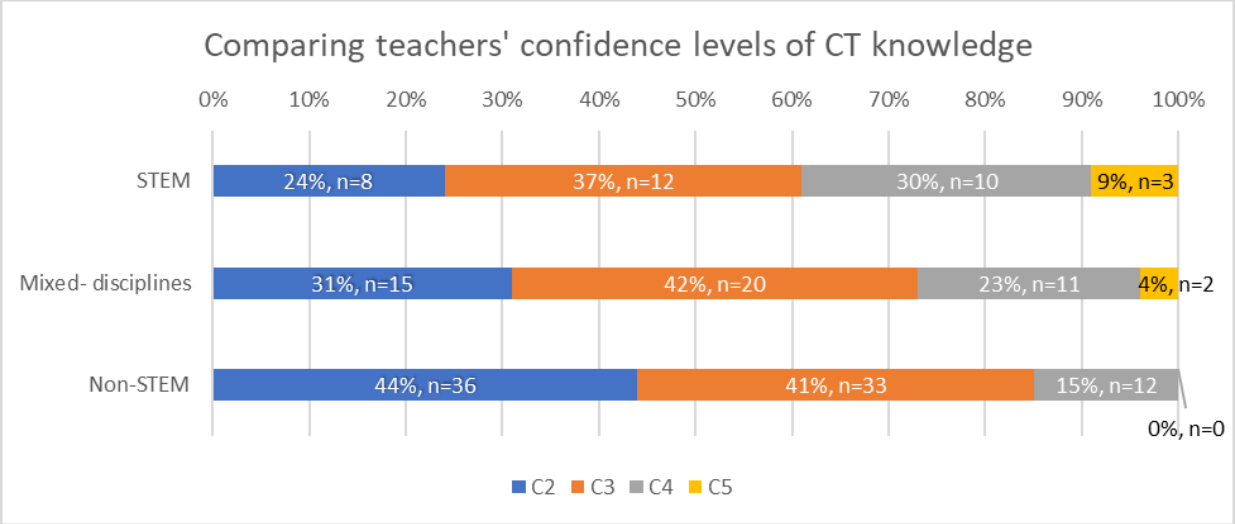


Figure 3. Comparing teachers' confidence levels

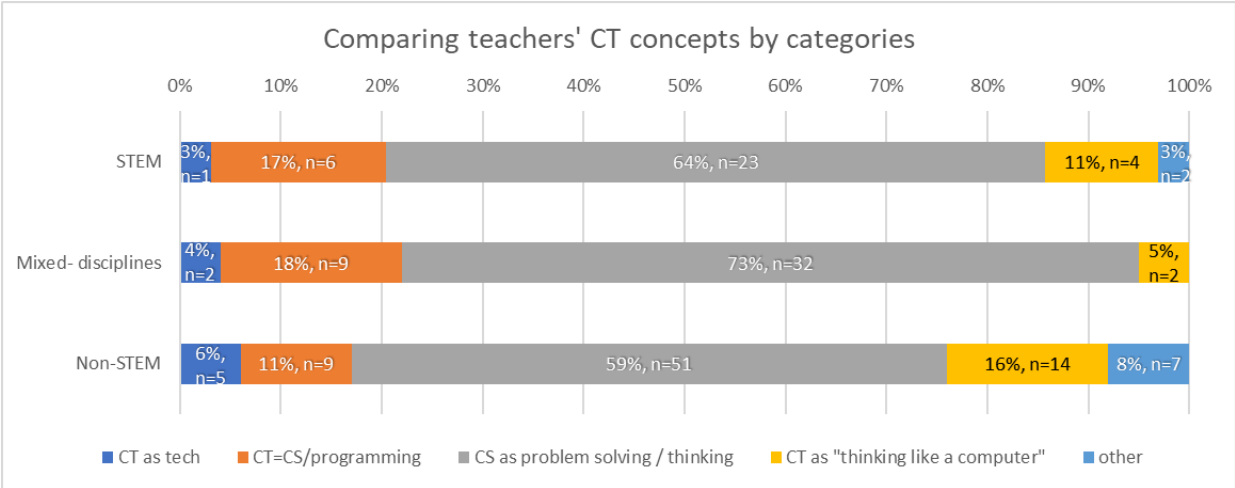


Figure 4. Comparing teachers' CT conceptions

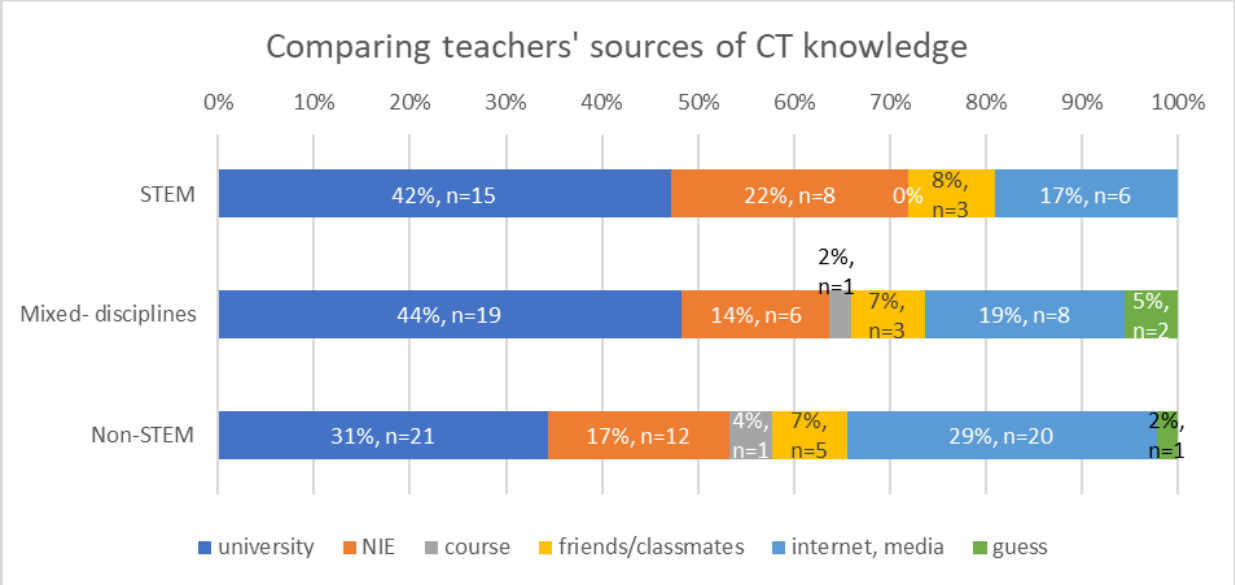


Figure 5. Comparing teachers' sources of CT conceptions

DOI: 10.34641/ctestem.2022.451

The Effect of Unplugged Programming and Visual Programming on Computational Thinking in Children Aged 5 to 7

Lisa BOSGOED[1*], Nardie FANCHAMPS[2]
[1]OBS De Graswinkel, Netherlands
[2]Open University, Netherlands

lisabosgoed@hotmail.com, nardie.fanchamps@ou.nl

ABSTRACT
This research focuses on the development of computational thinking (CT) among one-hundred and eight primary school pupils in the Netherlands aged five to seven years. It compares the use of unplugged programming and visual programming with on-screen output. In addition to the effect of using different programming environments, this research also establishes whether age differences and prior knowledge of programming have an additional influence. By means of a pretest-posttest design, using the validated quantitative instrument TechCheck, possible differences between the development of CT in both experimental groups and a control group could be objectively determined. To this end, pupils from both experimental groups have applied during five programming sessions of forty-five minutes each either unplugged story introduced smart games or used the plugged-in programming environment ScratchJr. Our results show a significant difference in CT development between unplugged programming and visual programming with on-screen output. Moreover, unplugged programming had a more positive effect on the development of CT compared to the control group than visual programming with on-screen output. A moderating effect could be attributed to age differences and prior knowledge of programming. This may provide an additional explanation regarding the identified impact and significant differences found.

KEYWORDS
Unplugged programming, computational thinking, smart games, primary education

2. INTRODUCTION
In recent decades, society has changed through various technological developments from an industrially oriented society to a mostly digitally focused knowledge community (Organization for Economic Co-operation and Development [OECD], 2008). To cope with this change, 21st century skills provide educational direction so that people can continue to develop in a focused way in order to function optimally. Computational thinking (CT) is an essential skill for making this transition. CT can be described as a set of problem-solving skills based on fundamental concepts from computer science and can be seen as a fundamental skill that is required in many everyday activities (Wing, 2006). The skill of CT can be promoted by different programming environments. However, little is known about the extent to which the differences and deviating characteristics of various programming environments can contribute to the development of CT skills (Brackman et al., 2017; Rose et al., 2017). We distinguish between a) plugged-in programming in which programming skills can be acquired by entering instructions and commands into a computer via graphical or tactile user interfaces using textual, visual or tactile programming languages resulting in on-screen output or tactile output; and b) unplugged programming where skills related to programming can be acquired without the use of a computer or digital processing agent. Results from previous research show that different design aspects of learning environments can have an effect on learning outcomes. For example, the extent to which the working memory is strained depends on prior knowledge and the way information is represented (concrete, iconic or symbolic) (Paas & Van Merriënboer, 2020). In addition, the children's development in each successive phase also plays a prominent role, from learning by physically manipulating perceptible objects to mental manipulation of more abstract or visual information (Sigelman & Rider, 2012).

3. PURPOSE OF THE STUDY
The aim of this study is to explore the effect that the type of programming environment and the associated characteristic differences have on the development of CT in young children. The research question is as follows: "Is there a measurable difference in effect on the development of computational thinking between unplugged programming and visual programming with on-screen output in children aged 5 to 7, controlling for age and prior knowledge of programming?"

4. METHOD
A quantitative, quasi-experimental study was conducted to determine the potential effects of the type of programming environment on the development of CT. Various schools were approached to participate in the study.

To determine the effect, a pretest-posttest design was applied. Children were non-randomly assigned into three research groups: unplugged programming, visual programming with on-screen output and a control group. Due to the COVID-19 pandemic, one school participated as the control group to reduce the number of contacts. As a pre- and posttest measurement, TechCheck was used as a validated instrument to determine the level of CT. As an intervention, children from both experimental groups were offered five programming lessons. Children from the control group participated in programming lessons after the study.

5. MATERIALS
To answer the research question, various unplugged smart games and ScratchJr, a plugged-in programming environment, were used to promote programming skills

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

(such as algorithms, loops and conditionals). All programming activities and games were carried out in collaboration. Children were offered one programming activity or game per lesson. Three-dimensional board games (e.g. Robot Turtles, Little Red Riding Hood and Sleeping Beauty) were used as unplugged smart games, where problems must be solved by applying sequential, manual steps (Brackman et al., 2017). In Robot Turtles, for example, players first need to arrange cards, which are included in the game, with written or pictographic commands such as "forward", "backward", "left", "right" and "jump". Then they have to move their turtle manually, according to the instruction, to receive a diamond. ScratchJr, as a plugged-in environment, is designed to teach young children programming within a two-dimensional environment (Rose et al., 2017). Instructions on the screen are created via graphical user interfaces by the drag-and-drop method. Instructions are created using blocks, which can be dragged from a library, that are pictographically displayed and represent commands. In the main program, these can be structured sequentially and in parallel. To apply a constructed instruction, the play button is pressed. ScratchJr offers various design aspects that allow children to create interactive animations, games and storylines. To determine the level of CT in the pretest and posttest, TechCheck was used. TechCheck has been validated in a group of 5- to 9-year-olds who participated in a study of visual programming with tangible output (Relkin et al., 2020). Results from the classical theory test and item response test show reliability and validity (α = .69). TechCheck measures CT as one construct using 15 multiple choice questions, which have a strong pictographic character. Furthermore, TechCheck do not distinguish between CT skills such as algorithmic thinking, problem decomposition or pattern recognition.

6. FINDINGS

Table 1 displays the results from the pretest and posttest. From this data it can be deduced that the posttest measurements show a higher average score and a lower standard deviation than the pretest measurements. Children from all experimental groups answered more questions correctly in the posttest than in the pretest. However, no significant differences were found between any groups $F(2.105) = 1.863$; $p = .160$. Comparing the averages (M) regarding the development of CT, the unplugged programming group had a higher mean score than the group that programmed using a visual environment with on-screen output and the control group.

Table 1. Means and Standard Deviations of CT

	Pretest	Posttest
Unplugged programming (n = 33)	11.48 (1.91)	12.21 (1.90)
Visual programming (n = 37)	9.05 (3.15)	9.08 (2.99)
Control group (n = 38)	11.32 (3.04)	11.42 (2.46)

After correcting means, significant differences were found between unplugged programming and visual programming with on-screen output, controlling for age (p = .008) and prior knowledge of programming (p = .042), as shown in Table 2.

Table 2. Means for Development of CT

	Before correction	Covariate age	Covariate prior knowledge
Unplugged programming	.73	.98	.90
Visual programming	.03	-.20	-.22
Control group	.11	.11	.20

Note. Covariate age groups unplugged: 5 years (n = 2), 6 years (n = 17), 7 years (n = 14); visual: 5 years (n = 12), 6 years (n = 18), 7 years (n = 7); control: 5 years (n = 7), 6 years (n = 20), 7 years (n = 11). Covariate prior knowledge unplugged: none (n = 0), few (n = 2), many (n = 31); visual: none (n = 9), few (n = 14), many (n = 14); control: none (n = 3), few (n = 2), many (n = 33).

7. CONCLUSION

Our research indicated that unplugged programming can play a prominent role in the development of CT, where age differences and prior knowledge of programming are of characteristic influence. In total, age has a moderate effect on the development of CT (η^2 = .09) and prior knowledge has a small-to-moderate effect (η^2 = .06). To generalise from our findings, more research is needed with larger groups.

8. REFERENCES

Brackman, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. *WiPSCE '17: Proceedings of the 12th workshop on primary and secondary computer education* (pp. 65-71). Association for Computing Machinery. https://dl.acm.org/doi/10.1145/3137065.3137069

Organisation for Economic Co-operation and Development. (2008). *21st century learning: Research, innovation and policy: Directions from recent OECD analyses.*

Sigelman, C. K., & Rider, E. A. (2012). *Life-span human development* (7th ed.). Belmont: Wadsworth.

Paas, F., & Van Merriënboer, J. J. G. (2020). Cognitive-load theory: Methods to manage working memory load in the learning of complex tasks. *Current directions in Psychological Science, 29*(4), 394-398. https://doi.org/10.1177/0963721420922183

Relkin, E., De Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and validation of an unplugged assessment of computational thinking in early childhood education. *Journal of Science Education and Technology, 29*(4), 482-498. https://doi.org/10.1007/s10956-020-09831-x

Rose, S. P., Habgood, J. M. P., & Jay, T. (2017). An exploration of the role of visual programming tools in the development of young children's computational thinking. *Electronic Journal of e-Learning, 15*(4), 297-309. https://doi.org/10.34190/ejel.15.4.2368

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. https://doi.org/10.1145/1118178.111825

DOI: 10.34641/ctestem.2022.463

Understanding Teachers' Attitudes and Self-Assessment Towards Computational Thinking

María ZAPATA-CÁCERES[1*], Nardie FANCHAMPS[2], Ibrahim H. YETER[3], Pedro MARCELINO[4], Estefanía MARTÍN-BARROSO[1]
[1]Universidad Rey Juan Carlos, Spain
[2]Open University, Netherlands
[3]Nanyang Technological University, Singapore
[4]TreeTree2, Portugal

maria.zapata@urjc.es, nardie.fanchamps@ou.nl, ibrahim.yeter@nie.edu.sg, pedro.marcelino@treetree2.org, estefania.martin@urjc.es

ABSTRACT
Around the world, attention is being paid to computational thinking (CT) in education. Integration into school curricula places additional demands on teachers, promoting the skills and attitudes necessary to teach and integrate CT into education. Above all, it is important that teachers themselves are aware of the importance of CT and have a clear perception of its meaning. To enable an effective and developmentally-enhancing implementation of CT in education, teachers must have competence to teach CT, recognize from what age CT can be taught, and how to transfer the acquired CT skills to other school subjects and areas. Therefore, we collected and compared data among schoolteachers from four different countries to enlighten their attitudes towards CT, their opinion about opportunities and possibilities for integrating CT into education, and how and from what age CT can best be applied. Furthermore, by administering and evaluating the Beginners Computational Thinking Test (BCTt), teachers' perspectives regarding this validated instrument for the assessment of CT are analysed. From qualitative data obtained, we could deduce information about teachers' self-assessment of competence, confidence, and motivation to teach CT. From quantitative data collected by administering the BCTt to teachers, we obtained indications of teachers' mastery of CT competence. The data analysis confirmed our hypothesis that discrepancies exist between teachers' self-assessment and their actual CT competence. It can be argued that the findings from our research, therefore, provide valuable information for further shaping teachers' future professionalisation concerning CT.

KEYWORDS
Computational Thinking, teachers' perspectives, primary education, attitudes, self-assessment

1. INTRODUCTION
Understood as the human problem-solving process that uses decomposition and requires thinking at multiple levels of abstraction (Wing, 2006), Computational Thinking (CT) is widely recognised as essential for coping with today's technological society (Shute, Sun, & Asbell-Clarke, 2017). The increasing attention to the development of CT in primary education compels teachers to adjust their teaching repertoire accordingly. Teachers are increasingly aware of the development potential that CT can offer both for students and to enhance their own teaching. But regardless of the perceived added value for education, the question arises as to whether teachers have a sufficient grasp of what CT is, what skills it encompasses, and how it can be used in practice, especially for subjects unrelated to technology or programming. This is according to a purposeful application of CT so that students can benefit from its use in the most transversal and optimal way. Such an approach and implementation in education requires that teachers be thoroughly equipped to become familiar with the underlying principles and characteristics of CT, yet insufficient attention has been paid to fostering the skills and attitudes needed to teach the new content (Mannila, Nordén, & Pears, 2018; Nouri, Zhang, Mannila, & Norén, 2020), and which type of guidance is most effective for teachers (Fanchamps, Specht, Hennissen, & Slangen, 2020). Moreover, teachers from different countries perceive that CT can foster a connection between different disciplines and provide an opportunity to support teachers' pedagogical practices (Diordieva, Yeter, & Smith, 2019). However, more research is needed regarding how CT can be integrated into a curriculum, on the pedagogical possibilities that CT can offer teachers, and on the required areas of professional development and teacher training.

Evidence suggests that teachers' understanding, prior knowledge requirements, pedagogical skills, knowledge of related technology, and self-confidence in teaching CT can be improved in a relatively short period of time through targeted professional training (Bower et al., 2017). Increasing student exposure to CT in schools is complex, requiring systemic change, teacher commitment, and the development of meaningful resources (Barr, Harrison, & Conery, 2011). With educational changes, teachers inevitably face such challenges. If teachers have inaccurate perceptions of CT, this will directly influence how they teach this area (Milton, Rohl, & House, 2007). Researchers have made strong connections between teacher efficacy and teacher behaviours that foster student achievement (Goddard, Hoy, & Hoy, 2000). If teachers do not feel effective in teaching CT, students may have negative learning experiences (Israel, Pearson, Tapia, Wherfel, & Reese, 2015).

The question is, however, whether and to what extent primary school teachers currently have sufficient insight into these underlying conditions. It is therefore particularly remarkable that much of the research conducted into the

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

possibilities and effects of CT focuses on students, but that there is still too little focus on the perception and awareness of the teachers. Differences in the situation within national curricula and between different countries also play a major role. Moreover, it can be stated that CT is the focus of attention in some countries, while this is much less the case in others. It is therefore valuable to know teachers' perspectives on CT for different countries. In order to make a representative comparison, we selected countries with different starting situations or levels in terms of their approach to education and research on CT (Saqr, Ng, Oyelere, & Tedre, 2021): Portugal, with a low level; the Netherlands, with an intermediate level; and Spain, with a high level. On the other hand, we have selected Singapore, as a non-European case with an intermediate level, given that the research carried out by Dagienė *et al.* on a total of 52 countries reveals that only 21% of the countries consulted include CT development in the school curriculum and of these, 91% belong to Europe (Dagienė, Jevsikova, Stupurienė, & Juškevičienė, 2021). By comparing the findings, indications can be obtained on the focus and underlying rationale for the importance of CT for each country. This comparison may subsequently contribute to the further definition and operationalisation of CT in education.

Regarding educational frameworks for teaching CT in primary education, one of the most cited in the literature and most empirically applied is the 3D framework (Brennan & Resnick, 2012). This framework divides CT into three dimensions: 1) computational concepts (concepts that programmers use); 2) computational practices (problem-solving practices that are produced in the programming process); and 3) computational perspectives (perspectives that designers form about themselves and the world around them). Besides, due to the recent introduction of CT in school curricula at an international level, there are still few validated instruments for the assessment of CT competence, particularly at early ages, Tang et al. identified 4 possible ways of assessing CT: traditional test, portfolio, questionnaires, and interviews (Tang, Yin, Lin, Hadad, & Zhai, 2020). In order to be able to analyse CT skills without relying on any particular learning environment, it is necessary to use a traditional test-type assessment tool such as the Beginner's Computational Thinking Test (BCTt), which is one of the few existing assessment instruments that have been validated in terms of reliability, under a psychometric approach, for early childhood and primary school students (Zapata-Cáceres María, Martín-Barroso, & Román-González, Apr 2021). The BCTt focuses on computational concepts and, partially, on computational practices, and has been included as an assessment instrument to be evaluated by teachers, as well as an element to assess teachers' actual skill competence in CT.

Qualitative and quantitative data were used to assess, from the teachers' perspective, aspects such as the importance of CT in the curriculum of each country, the training teachers receive in this area, the involvement of teachers in schools in different subjects, and the ages at which attention is paid to CT in schools. Data were also collected on teachers' perceptions of CT, their confidence and motivation to teach CT in their classrooms, and their competence self-assessment. Finally, we assessed the teachers' CT competence using the BCTt.

Considering previous rationale, our research question is: Are there specific differences between countries in terms of motivation, self-perception, knowledge, information, and competence in CT; as well as discrepancies in teachers' perception and actual competence in CT?

2. METHOD

To conduct the research, we designed an online survey that was administered to teachers (N = 328). Besides, the BCTt, targeted to children from 5 to 9 years old, was also included for teachers to complete. This approach has been chosen as such because, apart from collecting teachers' perceptions on CT, we also want to give them the opportunity to form an opinion regarding this validated assessment instrument and, at the same time, reflect on their own skills by completing the test.

The participants in this study were active pre-school and primary school teachers from 4 different countries: The Netherlands, Portugal, Spain, and Singapore; who were asked questions about the importance of CT, the methodology and pedagogy for developing this skill, and their self-perception of this competence. In a first approach, in which questions were asked about the level of teachers' knowledge of CT according to the subjects taught, 83 teachers from Spain, 54 from Portugal, 42 from the Netherlands, and 149 from Singapore took part. In a second approach, in which in-depth data were collected, 83 teachers from Spain, 54 from Portugal, and 42 from the Netherlands participated. This second phase was divided into the following blocks: 1) demographic data; 2) teachers' perspectives on the importance of CT and how to develop it at an early stage; 3) teachers' perspectives on how and when CT should be taught; 4) teachers' perspectives on how and at what age CT is taught in each school; 5) teachers' perspectives on how and at what age CT is taught in each country; 6) teachers' professional development and training in CT; 7) test administration (BCTt); 8) teachers' opinions and perceptions on the BCTt, and; 9) suggestions and comments. Answers are collected as options to select, Likert scales (1-5), or open text, depending on the nature of the question.

Finally, teachers answered the BCTt and were asked for their feedback. In addition, their test scores were collected for comparison with the teachers' self-perception of this competence. In this phase of the study, 83 teachers from Spain, 54 from Portugal, and 32 from the Netherlands participated. Moreover, the 149 teachers from Singapore were asked about their understanding of CT.

3. RESULTS AND DISCUSSION

In terms of demographic data, the teachers in the Spanish sample are younger than those in the other countries, with 70% being under 40 years of age, while in the Netherlands, only 40.5% are under 40 years of age and in Portugal this percentage drops to 13%. For this reason, in Portugal, more than 90% of teachers have more than ten years of study, while in Spain and Portugal only about

50% of teachers are so experienced. On the other hand, the percentage of women is higher in all samples, being the highest in the Netherlands (76%). All teachers are primary school teachers, and only in Spain are there also preschool teachers in the sample (13.1%).

As for the subjects taught by teachers, in Portugal, 68.5% teach computer science, technology, or programming, compared to 35% in Spain, 19% of the sample in the Netherlands (where 73.8% are classroom teachers), and none in the Singapore sample, where teachers were chosen to teach subjects not a priori related to the teaching of CT. It is noteworthy that 16.7% of teachers in the Netherlands sample teach children with special needs, compared to 11.1% in Portugal, 6% in Spain, and none in Singapore.

In addition to the greater teaching experience reported by Portuguese teachers, they are also perceived to have more knowledge, since 68.5% of the sample reported high or very high knowledge and skills in CT and only 9.3% low or very low knowledge (Likert scale 1 to 5). However, in Spain, only 29.7% report a high or very high level and an unexpected 32.1% a low or very low level, since they are much younger teachers and CT is a competence that is only recently being implemented in schools. In The Netherlands, the results are similar to Spain, with 21.4% of teachers with low or very low self-perceived knowledge of CT, and only 31% of teachers with a high or very high level (Likert scale from 1 to 5). As shown in Figure 1, it is noteworthy that in the Singapore sample, only 16.1% of the teachers who teach subjects that are not related to programming or technology indicate a high or very high level of knowledge, compared to 60.4% who have low or very low knowledge of CT, with 59 teachers out of the 149 sampled reporting no knowledge of CT at all.

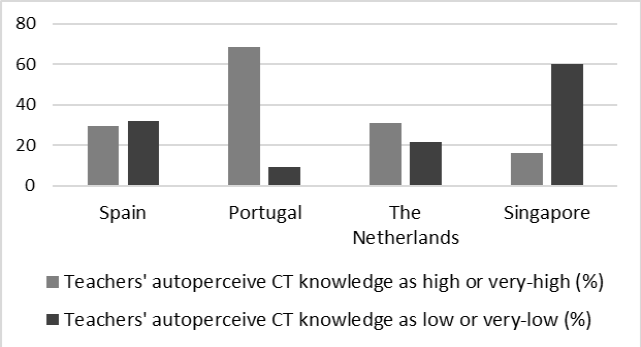


Figure 1. Teachers' auto perception of their CT skills.

In all samples, all teachers indicate that CT is important for students to be equipped to cope with the society of the future, and more than 80% consider it necessary for this competence to be integrated into the school curricula. However, Portuguese teachers are the most aware of the importance of incorporating this competence into the curriculum, with 94.4% considering it to be of great importance and more than 90% considering it to be related to student self-efficacy, compared to only 70% (approximately) of teachers in the Netherlands and Spain. In addition, more than 75% of the full sample and all teachers of children with special needs indicate that the CT can be very positive for these children.

A large majority of over 90% in Portugal and Spain are aware that CT can improve students' skills in other non-technology subjects, while in the Netherlands 76.2% are of this opinion and most teachers in Singapore believe that CT is only related to computer science. However, in Portugal and Spain, most teachers consider CT as a pedagogical mechanism and not only as an end in itself and understand that it can be taught independently of computer science as a cross-curricular competence. Moreover, also in Spain and Portugal, they advocate more teacher training in CT and teaching this competence in all schools (more than 90% of respondents approximately), compared to less than 70% of teachers of the Netherlands.

Regarding the second part of the questionnaire about the teachers' perspectives on how and when CT should be taught, the majority of teachers in all samples indicate that it should be taught later than age 4, and more than half of them think that it should be taught from age 7. Only around 17% believe that it should be taught before the age of 4, when it has been proven that it is better to start developing this competence as early as possible, similar to when learning a language (Mozelius & Öberg, 2017; Soosai Raj, Ketsuriyonk, Patel, & Halverson, 2018). Similarly, when teachers are asked whether CT should be taught in early childhood education, less than 60% say yes, and even in the Netherlands, only an unexpected 19% say it should be taught at this stage, when the integration of CT into the school curricula from the early childhood stage has long been promoted internationally. Moreover, teachers in all countries consulted believe that there are no gender differences in the learning of CT. However, previous research shows that girls are better at solving complex problems and boys at solving medium-difficulty problems (Eguiluz, Guenaga, Garaizar, & Olivares-Rodriguez, 2017; Guenaga, Eguíluz, Garaizar, & Gibaja, 2021), and there are also differences in the dispositions for the development of CT between boys and girls (Zapata-Cáceres & Martín-Barroso, 2021).

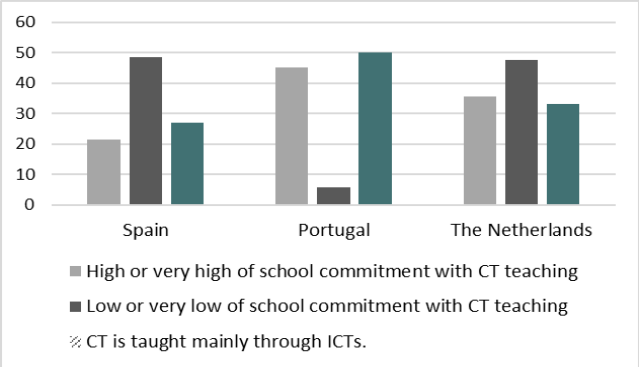


Figure 2. Schools' commitment to CT teaching.

Regarding block 3, teachers' perspectives on how and when computational thinking should be taught, there are large differences between the Portuguese sample and the rest. Portuguese schools are much more committed to the integration of CT in their classrooms and it is taught mainly through ICT-related tools (see Figure 2). However, teachers are not sufficiently informed as can be seen in Figure 3. Again Portugal has the highest percentage of schools, almost half, that include CT in their school

curricula, while in The Netherlands, more than half of the schools do not include it, and in all cases, there is a worrying percentage of teachers who do not know whether or not CT is included in the school curricula. In fact, teachers in all countries report a lack of knowledge and training on CT, materials, and resources.

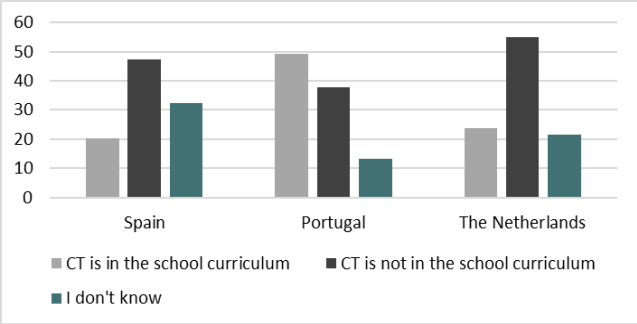


Figure 3. CT inclusion in school curricula.

When asked about how and at what age CT is taught in each country, teachers do not know how to answer the questions adequately because they lack information on the subject. In Spain and Portugal, the vast majority consider that, in their countries, CT is taught mainly from the age of 12 onwards, and that it is not a priority competence in education at national level. They also point to a lack of equipment, training, and resources for teaching CT, highlighting the lack of practicality of the initiatives that do take place. In the Netherlands, most teachers are aware that CT is taught at state level, but they also point to the lack of training (almost none), equipment, and time available to teach this competence. They feel that perhaps more attention should be paid to traditional subjects such as mathematics or language, rather than to transversal competencies such as CT.

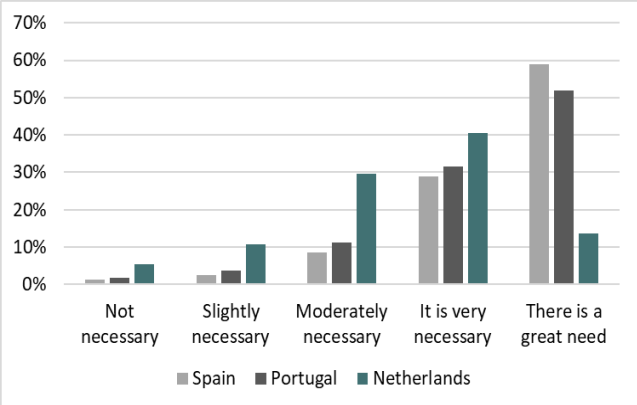


Figure 4. The need to include CT in the school curricula.

Although there are activities for teacher development and general training, few focus on CT, which is undesirable considering that it is a recent competence, and all educators need to be trained in it. In Spain, 44% of the respondents, and 35.7% in the Netherlands, have not received any CT training. Portugal is the country that is paying the most attention to this type of training, and only 18.5 % of teachers have not received any training. It is noteworthy that, although there is little training in CT in the Netherlands, the teachers surveyed are the least likely to perceive the need for such training (see Figure 4), with only 14% of teachers considering it to be a great priority.

Table 1. Correlation BCTt and Auto-assessment (AA).

		AA	BCTt total average
Auto assessment	Pearson Correlation	1	,241**
	Sig. (2-tailed)		0,002
	Sum of Squares and Cross-products	196,059	6,818
	Covariance	1,167	0,041
	N	169	169
BCTt total average	Pearson Correlation	,241**	1
	Sig. (2-tailed)	0,002	
	Sum of Squares and Cross-products	6,818	4,080
	Covariance	0,041	0,024
	N	169	169

***. Correlation is significant at the 0.01 level (2-tailed).*

In the third phase of the study, the BCTt was administered to teachers in Spain, Portugal, and the Netherlands. There is a significant correlation (Pearson's r = 1.00) between teachers' reported knowledge of CT (Mean = 3.25 on a Likert scale from 1 to 5), and the test score, as can be seen in Table 1, indicating that they are aware of their level of competence. It is noteworthy that 76.3% of teachers declare medium, high, or very high competence in CT. Furthermore, the ANOVA test shows no significant difference in test performance between the different countries ($F(2.166) = 1.958$, $p = 0.144$). Thus, teachers obtain similar results although, as seen above, there are large differences between countries in terms of the training received, the profile of the teachers, or their age and teaching experience.

Table 2. BCTt Average Scores by Country.

	Mean	*n*	*Std. Dev.*	*Min*	*Max*	*Sum*
Netherlands	21,75	32	3,733	7	25	696
Portugal	22,07	54	3,947	5	25	1192
Spain	20,80	83	3,879	6	25	1726
Total	21,38	169	3,896	5	25	3614

Table 3. BCTt total averages Gender

	Quantity				
	n	*M*	*sd*	*Median*	*Range*
BCTt Men	64	0.89	0.141	0.96	0.28-1.00
BCTt Woman	105	0.83	0.161	0.88	0.20-1.00

Note. n = respondents; M = average; sd = standard deviation

However, their overall performance on the test, i.e., their overall competence in the concepts associated with CT, is below what is expected and does not match their perceived competence, as the mean scores (considering the BCTt score as the sum of correct answers across the 25 test items) are lower than those obtained by primary school students in another research (Zapata-Caceres et al., 2020). It is remarkable that students aged 7 to 10 obtain an average score of 21.57 out of 25 on the test, which is very similar and even higher than the average score of 21.38 out of 25 obtained by teachers (see Table 2), especially in Spain, where teachers perform almost two points lower than primary school students. Although the samples are not statistically comparable, as the test is aimed at primary school students and is not validated to assess teachers' CT

competence (Zapata-Caceres et al., 2020), it would be expected and desirable that teachers would perform much better in the CT test than students, given that the test assesses a beginner level in CT. Maximum or close to maximum scores would be expected, since, in order to teach a subject, it is necessary to master it. The data also show large differences in test scores between teachers (*sd* = 3.896), some scoring unacceptably low (see Table 2). On the other hand, the data also indicate a significant difference (t (167) = 2.54; p = 0.015; CI [0.01-0.11]) between the performance of men and women, with the latter showing a worse performance (see Table 3).

Finally, teachers gave their opinion on the BCTt and made general comments on CT and its inclusion in the school curricula. In general, teachers found the test too difficult for primary school children, in fact, many teachers felt that children would not even be able to understand the questions at all. However, research suggests otherwise, and the test shows very high reliability for children aged 4 to 7 years, it was even necessary to create a more difficult test for children aged 7 to 10 years (El-Hamamsy et al., 2022) as a ceiling effect was observed. This indicates a discrepancy between the CT skills that teachers believe students have at an early age, and the level that children can actually achieve. Several teachers indicated that an oral explanation to the children would be needed before taking the test, which is indicated in the BCTt protocol.

In addition, many of the teachers do not understand exactly what CT is, especially teachers who do not have a computer-related background. For example, in the case of Singapore, where the entire sample is made up of teachers who do not teach computer science, technology, or programming, teachers are unable to define CT and some even indicate that they do not know the term. Those who do define it, relate it to algorithmics or computation, but do not find the implication that CT may have for the subjects these teachers teach.

4. CONCLUSIONS

Although most teachers have a high self-perception of their competence in CT, their actual skills in terms of the computational concepts related to CT do not match this self-perception, being much lower than expected, nor does their knowledge of the methodology to be applied or the age at which to start developing CT. One of the problems detected is that teachers largely underestimate children's ability in this competence at an early age and start teaching CT usually much later than would be advisable (Román-González, Pérez-González, & Jiménez-Fernández, 2017). Thus, CT teaching is mostly concentrated in students older than 7 years in all samples, which shows that not only teachers are not aware that CT teaching should start in early childhood education (especially in Portugal, the Netherlands and Singapore), but also that CT teaching is not implemented in schools until later. On the other hand, teachers who are not related to technology, computer science or programming have little information about CT and do not know how they could develop this competence in their subjects without using electronic devices, or how its development positively affects other areas away from computer science or technology.

As this competence is only recently being included in school curricula internationally, most teachers are not informed about its integration in their schools, nor do they know whether it is being taught in other schools or at the national level. The Netherlands teachers are the most informed about it, but surprisingly, they are the least willing to provide activities to develop CT.

Although there is training on CT, it is clearly not sufficient and should include: a) information on the appropriate starting age; b) competences that can be achieved by children at each age as well as training on the existing differences in terms of gender in developing CT, as well as in children with special needs; c) training on the transversality of CT, i.e. how to develop CT in different subjects, especially those not related to IT or technology (especially in Singapore), and without using electronic devices, i.e. CT unplugged (Brackmann et al., 2017; Zapata-Ros, 2019), especially in Singapore and Portugal; d) raising awareness of the importance of CT so that it is not seen as a waste of time and is perceived as a competence that positively influences the understanding and development of other subjects, especially in the Netherlands; e) training in CT competence, so that teachers are highly skilled and understand what CT is and the underlying computational concepts at an appropriate level to enable them to teach this competence to their students (especially in Spain); f) training in the incorporation of CT in both a transversal and a specific way, at school level and at state level in each country (especially in Spain, Portugal and Singapore).

In our opinion, the inclusion of training at all the levels described above, where shortcomings have been detected, is important and would improve both the quality and the content of the teaching of this competence in schools, adapting it to each age and characteristics of the students. This improvement would have a transversal impact on all subjects since the development of CT has a positive impact on other areas of knowledge.

It would be advisable to repeat this study in other populations since there are differences between countries regarding teachers' perception of CT. For example, in Singapore, more training is needed than in the rest of the countries regarding transversality of CT. In the Netherlands, there is a need for greater awareness of the relevance of CT. In Spain, more training is needed to understand the concepts around CT. In Portugal, although the sample was composed of older teachers than the rest of the samples, they are the ones who perceive more CT as a skill that needs to be developed to cope with 21st century society, but more training is needed on the transversality of CT and its teaching in non-technological subjects.

5. ACKNOWLEDGMENT

The authors want to thank the teachers that participated in this study and the STEAM Limburg group, The Netherlands. This work was supported by the AI.R-NISTH AI for Social Good Research Grant 2021, Nanyang Technological University in Singapore, and co-funded by the Madrid Regional Government, through the project e-Madrid-CM (P2018/TCS-4307), Spain, also co-financed by the Structural Funds (FSE and FEDER).

6. ETHICAL STANDARDS

The Ethical research board (cETO) of the Open University of the Netherlands estimates this research is in line with the rules and regulations and the ethical codes for research in Human Subjects (reference: U202111122/ManonQuint).

7. REFERENCES

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology, 38*(6), 20.

Bower, M., Wood, L. N., Lai, J. W., Highfield, K., Veal, J., Howe, C., et al. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education, 42*(3), 53-72.

Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. *Proc. of the 12th Workshop on Primary and Secondary Computing Education,* pp. 65-72.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proc. of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada, 1.* pp. 25.

Dagienė, V., Jevsikova, T., Stupurienė, G., & Juškevičienė, A. (2021). Teaching computational thinking in primary schools: Worldwide trends and teachers' attitudes. *Comp. Science and Information Systems*, 33.

Diordieva, C., Yeter, I. H., & Smith, W. S. (2019). Middle school STEM teachers' understandings of computational thinking: A case study of Brazil and the USA. In *2019 ASEE Annual Conference & Exposition*.

Eguiluz, A., Guenaga, M., Garaizar, P., & Olivares-Rodriguez, C. (2017). Exploring the progression of early programmers in a set of computational thinking challenges via clickstream analysis. *IEEE Transactions on Emerging Topics in Computing,* 1.

El-Hamamsy, L., Zapata-Cáceres, M., Barroso, E. M., Mondada, F., Zufferey, J. D., & Bruno, B. (2022). The competent computational thinking test (cCTt): Development and validation of an unplugged computational thinking test for upper primary school. *arXiv Preprint arXiv:2203.05980*.

Fanchamps, N., Specht, M., Hennissen, P., & Slangen, L. (2020). The effect of teacher interventions and SRA robot programming on the development of computational thinking. *CoolThink@ JC*, 69.

Goddard, R. D., Hoy, W. K., & Hoy, A. W. (2000). Collective teacher efficacy: Its meaning, measure, and impact on student achievement. *American Educational Research Journal, 37*(2), 479-507.

Guenaga, M., Eguíluz, A., Garaizar, P., & Gibaja, J. (2021). How do students develop computational thinking? assessing early programmers in a maze-based online game, 1-31. doi:10.1080/08993408.2021.1903248

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education, 82*, 263-279.

Mannila, L., Nordén, L., & Pears, A. (2018). (2018). Digital competence, teacher self-efficacy and training needs. *Proceedings of the 2018 ACM Conference on International Computing Education Research,* pp. 78-85.

Milton, M., Rohl, M., & House, H. (2007). Secondary beginning teacher's preparedness to teach literacy and numeracy: A survey. *Australian Journal of Teacher Education (Online), 32*(2), 37-56.

Mozelius, P., & Öberg, L. (2017). (2017). Play-based learning for programming education in primary school: The östersund model. Paper presented at the *European Conference on E-Learning-ECEL 2017, Porto, Portugal, 26-27 October 2017, 16.* pp. 375-383.

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry, 11*(1), 1-17.

Román-González, M., Pérez-González, J., & Jiménez-Fernández, C. (2017). *Which cognitive abilities underlie computational thinking? criterion validity of the computational thinking test.* doi: 10.1016/j.chb.2016.08.047

Saqr, M., Ng, K., Oyelere, S. S., & Tedre, M. (2021). People, ideas, milestones: A scientometric study of computational thinking. *ACM Transactions on Computing Education (TOCE), 21*(3), 1-17.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142-158. doi: 10.1016/j.edurev.2017.09.003

Soosai Raj, A. G., Ketsuriyonk, K., Patel, J. M., & Halverson, R. (2018). (2018). Does native language play a role in learning a programming language? *Proceedings of the 49th ACM Technical Symposium on Computer Science Education,* pp. 417-422.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). *Assessing computational thinking: Systematic review of empirical studies*, doi: 10.1016/j.compedu.2019.103798

Wing, J. M. (2006). Computational thinking test. *CACM Viewpoint*, 33-35. Retrieved from cs.cmu.edu/~wing/

Zapata-Cáceres, M., & Martín-Barroso, E. (2021). Applying game learning analytics to a voluntary video game: Intrinsic motivation, persistence, and rewards in learning to program at an early age. *IEEE Access, 9*, 123588-123602.

Zapata-Cáceres, M., Martín-Barroso, E., & Román-González, M. (2020). Computational thinking test for beginners: Design and content validation. *IEEE Global Engineering Education Conference (EDUCON'20),* pp. 1905-1914. doi: 10.1109/EDUCON45650.2020.9125368

Zapata-Ros, M. (2019). Computational thinking unplugged. *Education in the Knowledge Society, 20*, 1-29

 DOI: 10.34641/ctestem.2022.464

Digital Competence & Computational Thinking for Preschool Pre-service Teachers: From Lab to Practice

Ali HAMIDI[1*], Rafael ZEREGA[2*], Sepideh TAVAJOH[3*], Marcelo MILRAD[4*], Italo MASIELLO[5*]
[1,2,3,4,5] Linnaeus University, Sweden
ali.hamidi@lnu.se, rafael.zerega@lnu.se, tavajohsepideh@gmail.com, marcelo.milrad@lnu.se, italo.masiello@lnu.se

ABSTRACT
Digital competence is a skill associated with the 21-century abilities essential to contribute to today's and tomorrow's digital and technical environments. Computational Thinking (CT), which is a thought process for problem-solving, is one of the emerging trends that makes up digital competence. In our explorative study, we have used educational robotics with four pre-service teachers during their four-weeks placement at different preschools. We applied three distinct and complementary approaches to design and conduct this study: Systems Thinking (ST); Technological Pedagogical Content Knowledge; and Computing Pedagogy. Our findings are categorized in two main perspectives: pre-service teachers and children. In the pre-service teachers' perspective, the participants indicated that their educational program lacks specific content and activities related to digital competence, CT, and programming. Despite the initial pre-service teachers' thoughts on improvement of children's CT concepts, the findings show that CT practices such as collaboration and trial and error were developed. From the children's perspective, the empirical findings illustrate that digital competence and CT development vary depending on the age of the children; whereas logical thinking and pattern recognition are skills that were present along the whole age range of children (ages 2-6), other CT skills like algorithmic thinking were developed among older children only (aged 5-6). We learned that an ST approach can be helpful, as multiple factors are involved in the practice. It reveals the underlying features of the situation that emerge when components of the system interact with each other.

KEYWORDS
Computational thinking, systems thinking, digital competence, pre-service teacher education

1. INTRODUCTION
The so-called 21st century skills are an essential set of skills needed to be a competent citizen in our intensely technological society. To be able to contribute to it, we need to prepare our children to acquire those skills from very early ages. Along with different meta-skills such as problem-solving, collaboration, critical thinking, communication, and creativity, two other core skills make up digital competences, technical and information management (Van Laar et al., 2017). Having these skills means having the technical skills necessary to use digital technology and services and the knowledge necessary to find, analyze and critically evaluate information in different media, that is, media and information literacy. Computational thinking (CT) is a thought process as well as a skill for problem-solving and comprises different concepts and practices. According to Grover and Pea (2018), CT concepts include logical thinking, algorithmic thinking, pattern recognition, abstraction and generalization, evaluation, and automation. Regarding CT practices, they include problem decomposition, creating computational artifacts, testing and debugging, incremental development, collaboration and creativity.

The teaching of programming and CT in STEM related subjects has increased the need for digital competence development for pre-service and in-service teachers (Ottenbreit-Leftwich et al., 2021). The current trend is to teach digital competences to educators, but teacher training programs vary in terms of systematicity and in the various European countries (Bourgeois et al., 2019). However, little is known about an integrated approach for digital competence development for pre-service teachers (Howard et al., 2021), at least in Sweden. Most teacher education programs focus on content knowledge development rather than pedagogical intervention or technology knowledge around CT (Ottenbreit-Leftwich et al., 2021). Even less is known about preschool teachers' perceptions of CT and related professional development of this group of teachers (Papadakis & Kalogiannakis, 2022).

In Sweden, in 2018, the government revised the National School Curriculum with a new general section for compulsory schools emphasizing the importance of digital competences, both for teachers and students. Addressing the digital competences, CT is implicitly incorporated in terms of problem-solving and critical thinking. In fact, as part of the digital strategy of the Swedish National Agency for Education (Skolverket), CT skills are important to improve digital competence (Esteve-Mon et al., 2020). A recent study on 21st century skills from Sweden's pre-service teachers illustrates that digital literacy, critical thinking, and problem-solving are the most important skills they need for future teaching (Karakoyun & Lindberg, 2020). Therefore, this paper describes the different activities and outcomes of an exploratory pilot study named *Visual programming: From lab to VFU in preschool* within the context of a NordPlus funded project– *Digital Competence in Teacher Education in the Nordic Countries* (UIS, 2021). The project aims to establish a Nordic and Baltic network with the goal of promoting theory and practice in teacher education within the framework of digitalization in schools. The explorative study described in this paper took place during the Fall of 2020 and ended in June of 2021 with the participation of four preschool pre-service teachers exploring how to work with educational robotics. The theoretical approach of the study and the analysis of the data are based on the concepts of System

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Thinking (ST) as the meta-perspective for designing of the project and evaluation of the findings along with two others, namely Technological Pedagogical Content Knowledge (TPACK) framework and Computing Pedagogy.

The remaining of the paper is structured as follows: first, we describe the methodological approach of our work including the settings, theoretical foundations, and data collection methods. The empirical findings and discussions are then presented based on different data collection methods we used prior to and during the project. Lastly, the concluding remarks are described.

2. METHODOLOGICAL APPROACH

Different activities were conducted, including workshops, hands-on practices, visual programming sessions, and pre-service teachers' supervision both at university and preschools. The main goal of the different activities was to gain knowledge and to give an insight regarding the development of digital competence to the participating pre-service teachers, but also to create a long-term and sustainable collaboration between the research group and the university's teacher training program to further CT development into the program. The settings and participants of the project as well as theoretical frameworks of the study are explained below.

2.1. Settings

Four pre-service teachers (ages 23-32) in their fourth and final year of the preschool teacher program participated in the project during the Spring of 2021 (see Table 1 for details about participants and the children). Two PhD students and one research assistant in cooperation with two professor supervisors planned and designed the activities and guided the participating pre-service teachers through the different activities, as mentioned earlier. The whole internship period was four weeks. Firstly, the pre-service teachers spent two days at their preschool placement to get to know the children and their new colleagues. Then, they participated in 3-day workshops, conducted for 21 hours in total. During the workshops, the students had the chance to get acquainted with educational robotics construction kits that they would be using during their internship at the preschools. Also, the pre-service teachers were introduced to basic programming using a block-based graphical interface, and they planned the work to conduct at their internship placement at the preschools. Meanwhile, we conducted discussions with pre-service teachers to evaluate their understanding and perception of CT. Then, each of them took two sets of construction kits to use with the children. The pre-service teachers would later apply their experience from the workshops at the lab into their internship program in preschools for the remaining three weeks. Throughout the internship, the pre-service teachers shared their experiences and reflections with the rest of the group during three online meetings that were conducted via videoconference at the end of each week. For this project, we used the Engino Robotic Platform (ERP)[1] that includes building parts and visual programming software (KEIRO)[2] with the potential of dynamic construction.

Table 1. Participants Information

Pre-service Teacher ID	Children Age at Preschool	Number of Children
A	5-6 years old	15
B	2-3 years old	12
C	3-6 years old	16
D	3-5 years old	18

2.2. Theoretical Foundations

This explorative study was carried out based on the application of three distinct and complementary theoretical frameworks: ST, TPACK, and computing pedagogy.

Considering different factors involved in our study such as study objectives, pre-service teachers' background in terms of digital competence, and students with varying ranges of age, we perceived the situation as a system of integrated elements that needed a systems approach. So, we took the ST lens as one of the main theoretical frameworks for this study. ST helps us perceive the big picture, the boundaries, perspectives, and the relationships within the systems we work in (Cabrera & Cabrera, 2019). It plays a essential role in our approach, considering two points. Firstly, and from a broader perspective, ST leads towards the sustainable development of the educational context by taking an integrated holistic approach based on multi-stakeholder perspectives (Reynolds et al., 2018), which is aligned with the definition of the digital competence defined by Skolverket on critical and responsible usage of digital tools and resources (Feriver et al., 2019; Skolverket, 2021). Secondly, it helps to improve critical thinking and problem-solving abilities in complex settings through developing the metacognition skills of learners (Cabrera & Cabrera, 2019). A four-component ST model adapted from Cabrera & Cabrera (2019) is used for pre-service teachers' training. The model named *ST loop* includes plan and design based on our previous experiences (mental model), workshop training (approximation), practice in preschool (real world system), and feedback collection (information). This loop model guided our project and builds up our future research.

The TPACK framework (Mishra and Koehler, 2006) guided us in the training of the pre-service teachers during the workshops. Referring to the shortage in studies that focus on pedagogical aspects of pre-service teachers' CT development (Ottenbreit-Leftwich et al., 2021), we focused on pedagogical content knowledge (PCK) as well as technological content knowledge (TCK) of the TPACK model. As seen in Figure 1, CK presents digital competence and CT in our study. TCK refers to ERP sets and their programming software (Keiro) and PCK is applied through maker activities and storytelling.

[1] https://www.engino.com/w/

[2] https://enginoeducation.com/downloads/

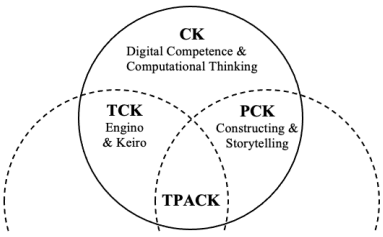


Figure 1. Applied TPACK framework

We have also followed a computing pedagogy model consisting of four implication's elements: approach, context, programming language, and engagement (Beauchamp, 2016; O'kane, 2019). It helped us design our activities in a way to maximize participants' engagement by engaging participants in making creative activities that are open-ended.

2.3. Data Collection Methods

We applied three data collection methods in our study: pre-questionnaire, interviews, and portfolios.

A pre-questionnaire was filled out online before starting the workshops. The goal of this survey was to get information about pre-service teachers' previous knowledge and experience on digital competence and CT. From the ST perspective, teachers are considered as important stakeholders throughout the design of our activities. In addition, looking at the part-whole notion in systems approaches (Cabrera & Cabrera, 2019), the pre-questionnaire gave us a view on constraints and issues (as parts) within the teacher program system (as a whole) in our study. The questions were designed according to the TPACK model in addition to some questions regarding CT key concepts.

At the end of each one of the four weeks of internships, we had meetings via videoconference to hear about the pre-service teachers' experiences and feedback using the ERP sets with the children at the preschools and to provide them with personal feedback. Feedback is an essential factor in any systems approach considering that different people have different priorities (Reynolds & Holwell, 2020). It is important to keep in mind that the four volunteers participating in this project worked with children of different ages, ranging from 2 to 6, and thus their children's cognitive development was quite diverse.

The pre-service teachers were expected to make individual reports of their weekly activities in the form of a *portfolio*. They were asked to include a description of their activities in the preschools, their own and children's experience around ERP and programming, and their achievements throughout the conduction of the different activities with the children. Students complemented the portfolios with pictures. Pre-service teachers who worked with younger children were suggested to focus on implementing CT concepts by applying methods based on storytelling. Storytelling method is applied when the concepts, definitions, and conceptualization of the provided contents are complex and could not be understandable for the target group (Wolz et al., 2011). This method is a teaching approach with the potential of creating and improving emotional intelligence to give children an insight into human behavior (Miller, 2021).

3. EMPIRICAL FINDINGS

According to the pre-service teachers' responses, digital technologies have not been properly introduced in their teacher program and there is insufficient specific education in the field of digital competence and programming for preschool teachers. Accordingly, this represented a considerable limitation when they were trying to apply digital tools and applications in their current programs' syllabus. Moreover, although the notion of digital competence is occasionally mentioned in their syllabus, the preschools' curriculum lacks sufficient content in connection to digital competence, CT, and programming. Our initial discussions with pre-service teachers in the first three days of the workshops suggest that the notion of CT was quite new to them. So, to evaluate pre-service teachers understanding of CT elements, we asked them to sort a few of the main CT concepts in order of importance related to children. Their responses showed that, in their view, logical thinking is the most important element of CT, then pattern recognition, abstraction, generalization, and lastly, algorithmic thinking. In addition, according to the pre-service teachers' responses, two concepts of CT, namely automation and evaluation, were the least important for children at the preschool level.

According to the interviews with the pre-service teachers when they had their internship with the children, they reflected on different aspects related to CT. One of the pre-service teachers who worked with children aged 3-6 reflected that the children of this age group can recognize patterns and follow instructions. The pre-service teacher pointed out that some of the children could figure out what the next steps were supposed to be. In addition, only one of the pre-service teachers managed to successfully make the children design simple codes, as these children were 5 to 6 years old. The other three pre-service teachers, whose children were younger (aged 2 to 4), mentioned that they had no expectations to do actual programming with children of these ages. Their empirical work with children at preschool revealed that the children were simply not mature enough to be able to understand the underlying concepts required to build an algorithm, and therefore they were not able to make any program. According to the pre-service teachers' reflections, trial and error efforts were common activities that the children did while building their models using ERP. Based on what they observed, the children they worked with applied very often trial and error when learning to join the different building pieces together, and they would engage in modifying the models they had built when they would not serve the purpose they had in mind.

Three pre-service teachers who worked with older children agreed that the children were able to collaborate with each other and build together. The pre-service teachers also commented that rather frequently, children were willing to help each other and solve problems together. An interesting aspect mentioned was that, in general, children under four years old seemed to have much more difficulties working

collaboratively with other children. On the contrary, children aged four and up showed a much better capability to work and build as a team and solve problems together. Pre-service teachers also referred to the concentration span of the children. That is, children sometimes lose focus and concentration during the activities.

The observations from the materials shown in the portfolios demonstrate that the children's skill levels and interests were different in terms of getting familiar with digital tools. Pre-service teachers believed that it was a worthy challenge when the children are trying to control the devices with mobile phones or tablets (see Figure 2). One pre-service teacher explained that it was important to let students think out-of-the-box so that they could build their own robots and follow their own plans.

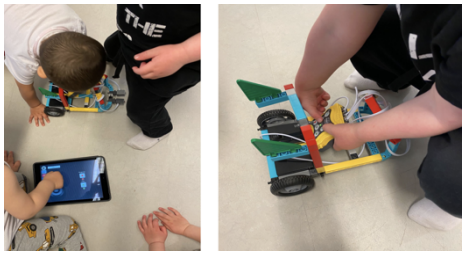


Figure 2. Controlling robots with a mobile /tablet

Pre-service teachers observed that children of age 5-6 tried to solve the problems by getting help from each other rather than asking the teacher. They mentioned that in general, the children were more motivated to build freely rather than building by following instructions. In addition, children wanted to build their own imaginary models and they collaboratively tried to figure out which components would have been suitable for their selected models such as a train. Based on collected content of the pre-service teachers' portfolios, the development of CT skills was observed when the children were trying to solve a common problem by using different components, changing the general structure, and modifying them based on their own imaginary stories. For example, as seen in Figure 3, the children imagined and discussed a familiar story about how they might make a train to carry their animal dolls on it. It is worth mentioning that children applied their previous experiences when they played with ERP sets. For instance, when assembling the ERP blocks, the children showed skills that were not demonstrated particularly when compared with playing with Legos' materials, which they had previously tried, according to the portfolios.

Figure 3. Redesigning the story and building a train

The pre-service teachers believed, according to the portfolios, that more teaching is required, having a special focus on more practical-oriented activities to improve pre-service teachers' learnings and skills. According to them, still it needs to be more transparent and specific when teaching each concept of CT. They also stated that the time they spent at the lab for learning and practicing CT concepts prior to their internship was not enough for sufficiently practicing with digital tools. Nevertheless, they expressed that the subjects of the use of digital tools in education and programming are necessary subjects that need to be included as an integral part of their preschool teacher program. The pre-service teachers stated that if their teachers or the people responsible for the teacher program are flexible, they could modify the workshops' plan and adapt it to the context in preschools to improve the teaching of CT.

4. DISCUSSIONS

From the triangulation of data, we learned that an ST perspective can be helpful as multiple factors contribute to the different CT practices. The children's age, the use of mobile devices, building skills, and previous experience of the children are examples of these factors in our study. According to the ST approach, we made a distinction about the results from two main perspectives: the pre-service teachers and the children one. Distinctions and perspectives are considered as two underlying fundamentals of any ST approach (Cabrera & Cabrera, 2018, 2019) and it helps us to frame the results in a simpler fashion.

4.1. Pre-service Teachers' Perspective

From the pre-service teachers' perspective, their educational program needs specific content and activities related to digital competence, CT, and programming. According to Skolverket's digitalization strategy, CT and programming must be integrated into Sweden's national preschools' curriculum (Skolverket, 2021). However, except for one of the participants, the rest of them were all unfamiliar with CT. According to Esteve-Mon et al. (2020), there is a two-way interconnection and a positive correlation between CT and digital competence. While CT builds up and integrates digital competence, students with a better perception of their digital competence gain higher CT skills. The participating pre-service teachers declared that they worked with digital technologies only occasionally during their studies, however, according to pre-questionnaire data, the notion of CT was quite new to them. That is why, improving their CT skills might influence their digital competence development.

Following the CK element of the TPACK model, we attempted to build their CT understanding through lectures and invited them to discuss CT concepts that were more relevant to preschool children. When considering the CT components *concepts* and *practices* (Grover & Pea, 2018), the pre-service teachers' initial perception was that CT concepts could be developed in early-age children. However, the results show that CT practices such as collaboration and trial/error were developed, according to pre-service teachers' observation. In line with Ottenbreit-Leftwich et al. (2021), lack of digital TPK in teacher

training programs is another aspect pre-service teachers mentioned in their workshops and interviews. Accordingly, we took advantage of the ERP affordances and storytelling-related activities as well as computing pedagogy approach to develop their TPK. In particular, PCK and TCK are useful to guide pre-service teachers on how to integrate CT in their activities, pedagogically and technologically (Ottenbreit-Leftwich et al., 2021). Regarding TCK, the existence of physical and virtual maker technologies associated with ERP and Keiro makes it possible to use physical construction for younger children who cannot handle actual coding. With PCK, we suggested digital storytelling by means of ERP to motivate students and engage them with the activity (Beauchamp, 2016; O'kane, 2019).

Storytelling has taken a supplementary role, whereby CT practices and TPACK teaching methodology provide a friendly learning environment. To do so, it needs to focus on the role of educators in terms of providing an interactive communicative learning environment. It is also in line with computing pedagogy approach (O'kane, 2019) we followed to increase the participants' engagement with the activity. As seen in Figure 4, pre-service teachers played different roles in a storytelling activity they carried out together during the workshops.

Figure 4. The practical activity of students playing the role of teacher and children.

4.2. Children's Perspective

Moving around the point from which an object is viewed is an essential characteristic of an ST approach, meaning that an idea can be the point or the view of a perspective (Cabrera & Cabrera, 2018, 2019). To deeply understand the digital competence and CT skills progress, we shift the perspective to children in this section. In other words, children are the lens through which we look at the CT and digital competence. Through the collected portfolios we could discern that pre-service teachers' work with the children was experienced differently depending on the children's age. Whereas logical thinking and pattern recognition are skills that were present along the whole age range of the children (ages 2-6), other CT skills, such as algorithmic thinking, were demonstrated and developed among older children only (ages 5-6). When younger children (ages 2-3) could not do actual programming, their CT skills could be detected and improved while participating in maker activities, controlling the robots using mobile devices (as a remote control), and using physical buttons to run the motors (see Figure 2). Giving the children simple guidelines and letting them try to figure out, for instance, how they can control their robots to reach a given goal is considered a way to introduce CT to young children (Papadakis & Kalogiannakis, 2022). This pattern was quite similar for the pre-service teachers who worked with the younger children. When it comes to coding activities of children aged 5-6, although they used simple functions to make a short program algorithm, it is very fascinating in terms of implementing the fundamental programming practices in this age group.

ERP provides an application for mobile devices and tablets that allow the user to control the robots by tilting the mobile phone, thus serving as a sort of remote control. This function helps the more curious children wanting to try to logically control the robots they built. The multifunctional potential of the robotic platform and the computing pedagogy approach we used in our study enables us to move around the TPACK framework from TCK to PCK (and vice versa) when pre-service teachers encounter limits according to children's ages. Except for one group of children (2-3 years old), the pre-service teachers observed that trial and error was the common CT practice among all children, leading to improvements during the construction process of the ERP. Many of the modifications of the constructions were made solely based on the ingenuity and creativity of the children. According to Grover and Pea (2018), trial and error reflects incremental development of students that is categorized under CT practices. The collaborative and communication skills, as well as a cognitive development associated with the ability to follow instructions among the children, are some aspects that show improvement by conducting activities of building with the ERP construction sets.

Considering storytelling as an interactive teaching method for children, following the patterns and scenarios while playing would be much easier than applying traditional teaching methods such as describing instruction and explaining concepts. This approach also gives children a chance to think outside-the-box with the potential of creating and improving emotional intelligence and provide support (Miller, 2021). The example illustrated in Figure 3, would be a kind of fully functional play story that provides an interactive educational environment where children come up with their own design strategy. That is like redesigning a story plan, but this time, it is made by the children's innovative thinking.

According to Reynolds and Holwell (2020), ST reveals the underlying features of the situation. For example, when pre-service teachers referred to the loss of concentration when younger children were participating in the building activities for more than 40 minutes, illustrating the importance of time management when working with children of this age.

5. CONCLUDING REMARKS

Currently, the development of digital competence through the integration of CT in formal and informal education is a growing trend in Europe and other regions of the world. In this pilot study, the researchers promote the development of digital competence and CT in preschool teacher education by using the Engino ERP construction kit.

Our findings indicate that there is a lack of professional development in preschool teacher education in terms of digital competences and CT. Integrating CT practices into an educational context would be a positive pedagogical and computational approach, where children are allowed to test, redesign play, change instructions, and decide whether to collaborate with peers. However, to improve CT skills, we need an ST approach and TPACK methodology as infrastructural development to both empower teachers as well as to provide a feasible CT learning environment in schools. Complementary methods of ST, CK, PCK, and TCK can be an effective approach to deal with very young children in terms of CT development. That is, when one approach is not possible to be applied due to the cognitive level of the young children, it can be supplemented with other methods. Our findings illustrate that not only CT practices are more developed than CT concepts for children aged 2-6, but CT development also vary depending on the age of the children. Whereas logical thinking and pattern recognition are skills that were present along with the whole age range of students, other CT skills like algorithmic thinking were developed among older children only.

Lastly, there is room for improvement in future activities in terms of bringing changes to the preschool curriculum and difficulties related to the Engino building and assembling system for preschool children as these building sets might not be the most suitable when working with children at a preschool level. Looking at the results of this paper on improvement of children's CT practices, more studies are needed to explore how to develop CT concepts.

6. REFERENCES

Beauchamp, G. (2016). *Computing and ICT in the Primary School: From pedagogy to practice*. Routledge.

Bourgeois, A., Birch, P., & Davydovskaia, O. (2019). *Digital Education at School in Europe. Eurydice Report*. Education, Audiovisual and Culture Executive Agency, European Commission. Available from EU Bookshop.

Cabrera, D. and Cabrera, L., (2019). Complexity and systems thinking models in education: Applications for leaders. *Learning, design, and technology: An international compendium of theory, research, practice, and policy*, pp.1-29.

Cabrera, D. and Cabrera, L., (2018). *Systems Thinking made simple.* 2nd ed. United States of America: Plectica Publishing.

Esteve-Mon, F., Llopis, M., & Adell-Segura, J. (2020). Digital competence and computational thinking of student teachers. *International Journal of Emerging Technologies in Learning (IJET)*, *15*(2), 29-41.

Feriver, Ş., Olgan, R., Teksöz, G., & Barth, M. (2019). Systems Thinking Skills of Preschool Children in Early Childhood Education Contexts from Turkey and Germany. *Sustainability*, *11*(5), 1478.

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, *19*, 1257-1258.

Howard, S. K., Tondeur, J., Ma, J., & Yang, J. (2021). What to teach? Strategies for developing digital competency in preservice teacher training. *Computers & Education,* 165, 104149.

Karakoyun, F., & Lindberg, O. J. (2020). Preservice teachers' views about the twenty-first century skills: A qualitative survey study in Turkey and Sweden. *Education and information technologies*, *25*(4), 2353-2369.

Miller, E. S. (2021, July). Designing Interactive Storytelling Games to Teach Computational Thinking. In *International Conference on Human-Computer Interaction* (pp. 342-351). Springer, Cham.

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers college record*, *108*(6), 1017-1054.

O'kane, L. (2019). *Computing pedagogy*. Icomput-uk. Retrieved February 3, 2022, from http://www.icompute-uk.com/news/computing-pedagogy/.

Ottenbreit-Leftwich, A., Yadav, A., & Mouza, C. (2021). Preparing the Next Generation of Teachers: Revamping Teacher Education for the 21st Century. In *Computational Thinking in Education* (pp. 151-171). Routledge.

Papadakis, S., & Kalogiannakis, M. (2022). Exploring preservice teachers' attitudes about the usage of educational robotics in preschool education. In *Research Anthology on Computational Thinking, Programming, and Robotics in the Classroom* (pp. 807-823). IGI Global.

Reynolds, M., & Holwell, S. (Eds.). (2020). *Systems approaches to making change: A practical guide.* London: Springer.

Reynolds, M., Blackmore, C., Ison, R., Shah, R., & Wedlock, E. (2018). The role of systems thinking in the practice of implementing sustainable development goals. In *Handbook of sustainability science and research* (pp. 677-698). Springer, Cham.

Skolverket (2021). *Curriculum for the compulsory school, preschool class and school-age education*. Skolverket. Retrieved January 10, 2022, from https://www.skolverket.se/getFile?file=3984

UIS (2021). *Digital Competence in Teacher Education in the Nordic Countries*. UIS. Retrieved February 10, 2022, from https://www.uis.no/en/teachers-professional-digital-competence.

Van Laar, E., Van Deursen, A. J., Van Dijk, J. A., & De Haan, J. (2017). The relation between 21st-century skills and digital skills: A systematic literature review. *Computers in human behavior*, *72*, 577-588.

Wolz, U., Stone, M., Pearson, K., Pulimood, S. M., & Switzer, M. (2011). Computational thinking and expository writing in the middle school. *ACM Transactions on Computing Education (TOCE), 11*(2), 1-22.

DOI: 10.34641/ctestem.2022.465

Computational Thinking in Flanders' Compulsory Education

Natacha GESQUIERE[1*], Francis WYFFELS[1]
[1]Ghent University – imec, IDLab-AIRO, Belgium

natacha.gesquiere@ugent.be, francis.wyffels@ugent.be

ABSTRACT
To modernise education, the Flemish government defined new learning goals that take account of 21st-century competences, in particular on 'digital competence and media literacy', of which 'computational thinking and acting' is one of the building blocks. Since September 2019, 'computational thinking and acting' has been compulsory in secondary schools in Flanders. The basic concepts decomposition, abstraction, pattern recognition and generalisation, and algorithm have been pushed forward. A closer look at the newly defined learning goals clarified that 'acting' is about basic knowledge in computer science and computational thinking practices. The learning objectives show that 'computational thinking and acting' is best addressed interdisciplinary in a socially relevant context. Based on the abundant scientific literature on the subject, we found these goals to fit into an international perspective. To support teachers, we are adjusting the teaching materials we already developed on physical computing, programming, and AI.

KEYWORDS
Computational thinking, compulsory education, K-12, Flanders, Belgium

1. INTRODUCTION
Computational thinking (CT) is a way of understanding and acting upon that digital world. A basic skill in CT enhances one's ability to understand and interact with technological developments, which can counteract the fear of technology (Bocconi et al., 2016). It is therefore not surprising that there is a growing focus on it in compulsory education all over the world. The same is true in Flanders. Introducing CT into compulsory education does not aim to turn everyone into programmers or computer scientists, but to acquire the skills of CT and to explore what they mean for the various disciplines (Barr & Stephenson, 2011; Hemmendinger, 2010).

The term 'CT' first appeared in an educational context in 1980, in Papert's book 'Mindstorms' and became popular when Wing launched it in 2006 as "a skill set for everyone" (Papert, 1980; Wing 2006). As the term 'CT' is fairly new, CT is not always known to teachers (Sands et al., 2018; Yadav et al., 2017). It is often mistakenly thought that the term covers a whole new range of subject matter, when in fact the concepts of CT go way back, to a time when computers did not yet exist (Denning & Tedre, 2019).

According to Lowe & Brophy (2017), almost every digital system is part of a human-centred system. Wing (2006) argued that CT is about "understanding human behavior". After all, digital applications are made by and for people. So, CT is also about people (Curzon & McOwan, 2017; Denning & Tedre, 2019). Not everyone gives the same meaning to the term 'CT'. Also, the way in which someone applies CT differs according to what they want to use it for (Hemmendinger, 2010; Weintrop et al., 2016). Moreover, the content of the term is constantly changing along with developments in computer science (CS), i.e. with 'what modern computers can do' and the further digitalisation of society (Denning & Tedre, 2019). CT is mainly about the ability to "effectively use a computer to solve the complex problems that people face" (Lu & Fletcher, 2009). Although there is no consensus on a definition of CT, examining the scientific literature to see what the various descriptions of CT have in common led Selby and Woollard (2013) to the following definition: "CT is a focused approach to problem solving, incorporating thought processes that utilise abstraction, decomposition, algorithmic design, evaluation, and generalizations". Here, generalisation refers to pattern recognition. Although CT is required to perform activities, such as automation, and modelling, they are not included in the definition, to distinguish between CT concepts and CT practices. Over the years researchers formulated many definitions of CT. Concepts were added and others disappeared. Concepts of CT like 'decomposition', 'generalisation and pattern recognition', 'abstraction', and 'algorithm' appear to be broadly agreed upon (Bocconi, 2016; Grover & Pea, 2017; Lodi, 2020; Lowe & Brophy, 2017).

Digital skills are seen as a way to acquire and evaluate the 21st-century skills. This is the main reason why countries are introducing CT into education. The 21st-century skills are general-purpose skills that should enable us to be resilient in a rapidly changing society. These skills are considered necessary to be able to function in and contribute to today's society: communication, cooperation, digital skills, social and cultural skills (including citizenship), creativity, critical thinking, problem solving, and productivity. Some include self-regulation in this list. Not all these competences are typical of our time, but due to the increasing presence of technology and digitalisation, 21st-century competences have gained in importance. Because of the rapid developments, digital skills should not only be limited to the use of applications but should also include, in addition to ICT skills, information and data literacy, CT, and media literacy (Bocconi et al., 2016; Denning & Tedre, 2019; Thijs et al., 2014; Voogt & Roblin, 2012; Voogt et al., 2013). The European Union emphasises that digital competences "involve the secure, collaborative and creative use of ICT, including coding" (European Union, 2015). "Skills, such as problem solving, critical thinking, ability to cooperate, creativity, computational thinking, self-regulation are more essential than ever before in our quickly changing society" (European Union, 2018).

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Many researchers see CT as a skill that can be addressed in all subjects and provide numerous examples of this (Barr & Stephenson, 2011; Grover & Pea, 2017; Yadav et al. 2017). Hence, many voices are heard calling for CT to be offered cross-curricular, in relevant real-world contexts, especially given the link with 21st -century skills. But there appears to be a gap between the importance attached to the 21st-century competences and how they are dealt with in practice in the schools, f.i. little time is spent on them, the objectives are narrowed down to the use of software only, no real-life contexts are used or the opportunities that the digitalisation offers for learning differently remain unused. Examples of how 21st-century competences, especially digital competences, can be addressed within familiar lesson content can help (Voogt et al. 2013; Goldberg et al., 2013). And although one does not always need a computer to acquire certain skills of CT, the focus should ultimately be the computer (Barr & Stephenson, 2011). After all, digital skills, including CT, get so much attention because they are the means to make us function in today's (digitalised) society and understand the impact of digitalisation on society. The importance the European Union attaches to programming stands out. Many consider programming a crucial skill. Guzdial (2015) considers programming indispensable to be computationally literate: "Achieving computational literacy in society means that people can read and write with computation, which includes an ability to read and write computer programs". Some rightly warn against seeing CT as too separate from the computer (Denning & Tedre, 2019; Lodi, 2020). If CT is taught in non-computer science classes, or for example through unplugged activities, then the link to the computer should be made explicit to achieve the transfer of the activity to an understanding of CS. Bell and Vahrenhold (2018) argue that the popular 'CS Unplugged' is best linked to contemporary technology. 'Unplugged' often makes complex concepts more accessible, but they are not sufficient for learning to think computationally. To fully understand digital systems, programming, e.g., will have to be involved as well.

For several decades CS was almost absent in the curriculum of Flemish schools (wyffels et al., 2014). In 2014 the Royal Flemish Academy of Belgium for Science and the Arts brought out a report to call for action on CS in compulsory education. They link CT inextricably with CS as they describe CT as

the human ability to solve complex problems using computers as a tool … It is the process by which aspects of computer science are recognised in the surrounding world and applying computer science methods and techniques to understand and solve problems in the physical and virtual world (Samaey et al., 2014).

The Academy sees CT as a form of problem solving but with an interpretation that "computational thinking results in a computer program or a robot that really works". The report was one of the reference frameworks for the reform of Flemish secondary education concerning digital skills. The learning objectives of digital skills were expanded to include the concepts of CT and the basics of CS. Finally, Flanders follows the international trend of making CS education compulsory.

In the remainder of this paper we discuss the new learning goals in Flanders (section 2). We briefly mention how schools implement CT (section 3). We describe how CT is introduced (section 4), how CT goals are intertwined with other learning goals (section 5), and how this is related to our own work (section 6).

2. NEW LEARNING GOALS

In Flanders, the government recently formulated new learning goals for secondary education, with the aim of a future-oriented education that considers the challenges of the 21st-century. These learning goals list the minimum learning objectives schools must achieve with their pupils and explicitly mention the expected factual knowledge and the corresponding conceptual, procedural, and metacognitive knowledge. In addition, the proficiency level of the learning objectives is specified according to the revised Bloom taxonomy. The newly defined learning objectives are divided into 16 key competences, such as 'Competences in Dutch', 'Socio-relational competences' 'Civic competences', 'Digital competence and media literacy', 'Learning competences' and 'Competences in mathematics, science and technology'[1]. The starting point of the key competence 'Digital competence and media literacy' is "going into the digital developments and the importance of basic knowledge and good use of ICT to be able to participate in society". This key competence is composed of 3 building blocks: 1. 'Digital media and applications to create, participate and interact' around the use of information and communication technology. F.i., the use of online tools, creating digital content, and digital citizenship. This block is linked to learning objectives on acquiring and processing information from the key competence of learning. It addresses digital developments and the importance of basic knowledge and good use of ICT to participate in society. In the formulation of the learning goals in this block, there is an explicit reference to creation, sharing, and collaboration, a reference to the 21st-century competences. 2. 'Computational thinking and acting', which aims to provide "a basic knowledge and skill of computing", and to promote problem-solving thinking. 3. 'To deal responsibly, critically and ethically with digital and non-digital media and information', which is about media literacy. This block treats the impact of technological development on society and the ethical aspects associated with it. It also aims to reinforce critical thinking. F.i. the learning outcomes formulated for this building block pay attention to image literacy, which is important for dealing critically with various media. Given the interconnectedness of the digital world with all of our lives, there is no doubt that 'Digital competence and media literacy' cannot be separated from the other 15 key competences[2].

In short, since September 2019, the new learning goals were introduced in the first year of secondary school. They will be further implemented in the other years of schooling year by year. They represent a measurable standard as a

[1] https://onderwijsdoelen.be/uitgangspunten/4647

[2] https://onderwijsdoelen.be/uitgangspunten/4814

basis for curricula to be developed. The curricula and the evaluation of the learning progress of the pupils have to be adapted to this standard, but the government does not dictate how schools should achieve this. Most schools, however, receive guidelines from umbrella organisations that provide curricula and advice on pedagogical approaches.

3. TEACHING CT

Since September 2019, 'computational thinking and acting' has been part of the compulsory curriculum for all pupils in secondary education. For pupils in middle school (the first stage of secondary education), the subject has even been given the status of basic literacy. Each individual pupil must achieve this set of objectives. Although the government initially wanted to force CT to be taught in all subjects (sciences and non-sciences), this idea was abandoned[3]. This is a missed opportunity, f.i. to offer CT integrated with other 21st-century competences.

How 'CT and acting' is or will be dealt with in Flemish schools is still a big question mark. Schools can freely decide how they plan to achieve the learning goals of 'CT and acting', as long as the pupils meet the expected knowledge. Some school directors choose a project-based approach and want to see CT in integrated STEM lessons. In other schools, it is taught as a separate subject, which does not always benefit the interdisciplinary aspect and the link with society. Another approach is to offer it in project weeks. Some directors leave it to the teachers to find their own way. Schools do gratefully make use of initiatives related to CT that they can bring to school, such as the international Bebras competition, EU CodeWeek, the Belgian initiatives WeGoSTEM and AI Op School[4].

4. DEFINITION OF CT AND EXAMPLES

To make clear what is meant by 'computational thinking and acting' the minimal learning goals come with a definition given by the Flemish Government:

By computational thinking and acting we understand a process in which one arrives at output using the following techniques: recognising patterns (pattern recognition) and generalising (generalisation), dividing a problem into sub-problems (decomposition), abstracting the data or the problem itself (abstraction), shaping the solution method (modelling) and following a fixed step-by-step plan (algorithms)[2].

In addition, the Flemish Government clarifies:

These skills, found in computer science, help students to get a better overview of complex problems. Understanding these concepts helps to understand how a computer works and, at a later stage, to use the computer as a tool to solve a problem. Knowing the basic concepts and functions of computers and computer networks and being able to name, install and operate hardware and software are basic requirements for acquiring and processing information digitally, communicating and sharing and creating content.

By "not only teaching pupils to use digital technology, but also to understand how it works", the aim is to prepare them for life in a rapidly changing world and to equip them to think critically about the impact of technology on privacy, employment, and health. So, what was envisaged ties in with the need for digital literacy to acquire 21st-century competences.

To illustrate, some of the learning goals for secondary schools on 'computational thinking and acting':

For middle school: "The pupils distinguish building blocks of digital systems. (understanding)" and "Pupils apply a simple self-designed algorithm to solve a problem digitally and non-digitally. (analysing)". Depending on the field of study, followed by those in the second grade (level 9-10): "The pupils explain how building blocks of digital systems relate to and interact with each other. (understanding)". And "The pupils solve a defined problem digitally by adapting an algorithm provided. (creating)" or "The pupils design algorithms to solve problems digitally. (creating)". Depending on the field of study, in the third grade (level 11-12) followed by: "The pupils assess building blocks of digital systems in terms of their own use and their use in a social context. (evaluating)". And "The pupils solve a complex problem digitally by adapting an algorithm provided. (creating)" or "The pupils program solutions to problems using self-designed algorithms according to a certain system. (creating)".

The knowledge that pupils have to acquire with regard to 'CT and acting' includes: decomposition, pattern recognition, abstraction, algorithm, digital representation of information, testing and debugging, modelling and simulation, principles of programming languages (sequence, loops, selections, variables, data types, operators, functions), input-processing-output, binary representation, hardware, data format, applications such as word processing and games, operating system, communication between digital systems, properties of connections such as bandwidth, safety, reliability, connection between analogue and digital representation, internet, and impact of algorithms.

Hence, in Flemish education, the following concepts of CT have been pushed forward: decomposition, pattern recognition and generalisation, abstraction and algorithm. The new learning goals fit in nicely with the known consensus but go beyond the four basic concepts of CT. They also aim at basic knowledge of CS, since computational practices like modelling, testing and debugging, digital representation of information, and principles of programming are included.

In addition to the new term 'computational thinking', it is not always clear to teachers what is meant by 'decomposition', 'pattern recognition', 'abstraction' and 'algorithm'. For example, the term abstraction also occurs in mathematics. However in mathematics, abstraction does not exactly mean the same thing as in a computer-related context. In the aforementioned definition of the Flemish Government on CT, these new terms only briefly occur: decomposition is dividing a problem into sub-problems,

[3] https://etaamb.openjustice.be/nl/decreet-van-12-februari-2021_n2021031270.html

[4] https://www.aiopschool.be and https://www.dwengo.org

and algorithm is about following a fixed step-by-step plan. Abstraction, pattern recognition, and generalisation are not even explained. In the literature, researchers clarify each of these concepts separately and describe how one can work with them in a classroom (Bell & Vahrenhold, 2018; Dasgupta & Purzer, 2016; Grover & Pea, 2017; Rich et al., 2019; Statter & Armoni, 2016; Yadav et al., 2017).

An example of a 'basic literacy' goal (one that every pupil must achieve) in middle school: "The learner demonstrates in functional contexts basic skills to create and share digital content". The conceptual and procedural knowledge accompanying this goal is: "Digital media and applications to create and share digital content, such as online and offline word processing, calculator app, digital image processing, graphic programming language, browsers, electronic mail, common social media applications, cloud applications". This goal from the first building block of 'Digital competence and media literacy' is connected to 'CT and acting' and connects digital competences to creativity. The CT practice 'programming' is encouraged to be used to foster creativity. In any case, programming offers pupils an opportunity to express themselves in a creative and contemporary way, f.i. by implementing an original solution to a problem or by creatively creating a digital system. The computer also offers new possibilities for creative expression. For example, utilising digital tools or by coding, creative solutions can be realised which formerly were impossible. Mishra and Yadav (2013) argue that "human creativity can be augmented by CT, in particular with the use of automation and algorithmic thinking", and that CT can transform users into creators. Given the importance attached to creativity as a 21st-century competence, it is useful to consider how CT can best find its place in the curriculum in order to promote creativity in pupils (Voogt et al., 2015).

Grover and Pea (2017) note that certain aspects of CT overlap with the 21st-century competences of collaboration and creativity and believe that CT combined with "other modes of critical thinking" can serve to address the challenges of this century. Earlier, Papert (1980) linked CT to competences such as problem solving, collaboration, creativity, and communication. Gretter and Yadav (2016) state that, within the digital skills, media and information literacy are complementary to and partly overlap with CT. Both are about "the importance of being digitally literate as seen from the broader, social impact of the Internet". This brings an opportunity to teachers to offer these skills to pupils in an integrated way across different subjects. CT can reinforce other 21st-century competences, including media literacy, critical thinking, citizenship, and cultural awareness. One of the new goals on media literacy in the 2nd grade (level 9-10) is "The pupils explain the mutual influences between the individual on the one hand and media, digital infrastructure and digital applications on the other hand". This topic can be addressed via the impact of algorithms in daily life and illustrates the overlap between CT and media literacy. On the other hand, placing too much emphasis in lessons about CT on the acquisition of skills such as perseverance, the ability to work together and dealing with complexity and ambiguity, could lead to working on the skills of CT itself being lost (Voogt et al., 2015).

5. CT IN OTHER KEY COMPETENCES

Generally, as we live in a knowledge society, digital competences consist of basic ICT skills, information literacy, CT, and media literacy. In the new Flemish learning goals, CT, basic ICT skills and media literacy are part of the same key competence, reflecting this.

In addition to the learning outcomes in the key competence 'Digital competence and media literacy', there are other learning goals related to digital skills, and in particular to CT. In the format of the new Flemish learning goals, information literacy is not included in the key competence of digital competence; information literacy is included in the key competence of learning. The key competence of learning has four building blocks from which two include learning goals on CT, namely the blocks 'Use appropriate (learning) activities, strategies, and tools to acquire, manage and process information critically, digitally, and non-digitally, considering the intended learning outcome and process' and 'Recognising a (research) problem and finding an answer or solution using appropriate (learning) activities, strategies and tools'. An example of a learning goal from the former block is "The pupils process digital and non-digital information from various sources in a strategic manner into a coherent and usable whole". The latter tackles problem-solving thinking, to which CT is closely related. This block contains learning goals that require a digital skill, such as "The pupils carry out an investigation technique to acquire digital and non-digital data based on a research question". In the learning goals, overall, much attention is also paid to problem solving and communication. The knowledge that must accompany this includes decomposition, formulating problems and generating ideas, designing and programming algorithms, collecting data, making measurements, evaluating and adjusting, applying an iterative process, and applying computational skills.

Kafai and Proctor (2022) frame CT within "computational literacies in the 21st-century". They emphasise that it is not just about technical skills, processing algorithms and information, and being able to program, but also about the social and cultural dimensions that go with it. It is also about citizenship, critical use, personal expression, and connecting with others. Working with computers is also a social activity where one has to take into account its role in society; an example of this is dealing with cultural bias in computer systems. They also caution to carefully choose contexts wherein CT is offered (one does not want to inherit the leaky pipeline from STEM), ensuring that the target group for whom teaching materials are being created is engaged.

Moreover, some of the learning goals within the key competence 'Competences in mathematics, science and technology', especially the ones relating to mathematical skills and integrated STEM, lend themselves well to the acquisition of digital and computational skills and build-up from the first to the sixth year of secondary education. These goals range from honing the use of ICT tools to

maintaining a computer. Table 1 shows some of the contexts these learning goals are situated in.

Table 1. Learning Goals from other Key Competences.

Digital applications	- creating, sharing, collaborating, communicating, participating and interacting with digital applications - ICT in certain school subjects
(Digital) systems	
Information literacy	- acquiring/processing information - data literacy - modelling and simulation
Media literacy	
Algorithms	
Logic	

In the key competence 'Competences in mathematics, science and technology' one of the goals in the 1st grade is: "The pupils investigate the principles of construction and operation of technical systems, their subsystems and components as well as their mutual interrelationships in the context of a technical process", which tackles system thinking. The knowledge accompanying this goal includes the function of sensors and actuators, logic, and input-processing-output. An example on integrated STEM in the 2nd grade: "Using concrete social challenges, the pupils explain the interaction between the individual STEM disciplines and between STEM disciplines and society", of which the knowledge includes system thinking. Two examples of modelling and simulating in some fields of study in 3rd grade are "The pupils use models for exponential growth" and "The pupils work out models and simulations using simulation software". Weintrop et al. (2016) developed a framework useful for teachers to work with CT in mathematics and science.

CT and 21st-century skills are interwoven in the new learning goals. The 3rd grade goal "The pupils critically process digital and non-digital information from various sources into a coherent and usable whole, taking into account possibly contradictory information", links digital competence, to critical thinking and information literacy. A goal for 1st grade shows the link between ICT skills and communicating and cooperating skills: "The pupils demonstrate basic skills for working together communicate and participate in initiatives in a digital way".

6. WORK IN PROGRESS

In Flanders, there is a lack of adequately trained teachers (Bocconi et al., 2022). Since they are crucial to making the integration of CT successful, teachers need help to develop curricula and to bring CT in their familiar lesson content. Examples can familiarise teachers with the new concepts (Grover & Pea, 2017; Voogt et al 2015; Yadav et al., 2017). Based on the knowledge that the literature brings and our own experiences, we want to address the needs of teachers and teacher trainers in Flanders regarding CT. We want to give teachers insights into what CT is, what the basic concepts entail, and teach them to recognise when it is opportune to introduce basic concepts of CT in their lessons. Therefore, we are adjusting the teaching materials we already developed on physical computing, programming, and AI: we add concrete examples of how to integrate 'computational thinking and acting' in interdisciplinary and school subject-related contexts and clarify the terminology used. These open-source, online materials will be available to professionalise Flemish teachers in CT. We want to provide both unplugged and plugged activities, starting from day-to-day examples to solving complex problems. We will try to elucidate the different levels of abstraction, the different ways of decomposition and pattern recognition, and how to address algorithms in an unplugged and plugged way. We are developing a frame for evaluation, considering the Bloom taxonomy. For this, we can rely on the work of Selby (2015) or Bell and Vahrenhold (2018). Within the framework of an ongoing project, we test our material in schools and adjust it based on the feedback from teachers.

7. CONCLUSION

Taking into account the 21st-century competences, the Flemish government imposed new learning goals, CT included. The reason for the new learning goals is to be found in the need for 21st-century competences to be able to function in our digitalised society, which is in line with how CT is viewed internationally. Teachers must not lose sight of this higher goal of the new learning objectives. In compulsory education, in Europe and elsewhere in the world, there are still plenty of questions on how the 21st-century competences can be integrated into the existing curriculum or how they can be developed cross-curricular. The same question is posed considering CT. In this paper we demonstrated how, within the frame of the Flemish government, the CT learning goals are connected to the ones about developing the 21st-century competences. We also discussed some noticed connections between CT and STEM. These links show Flemish teachers the way how to address these new learning goals in an integrated manner, such as in an interdisciplinary and socially relevant context. For the time being, in Flanders the way teachers deal with 'CT and acting' varies from school to school.

8. ACKNOWLEDGEMENTS

The work in this paper has been funded by a grant from the Flemish Government for the implementation of a pilot project to strengthen teacher education.

9. REFERENCES

Barr, V., & Stephenson, C. (2011). Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community? *ACM Inroads*, *2*(1). 48–54.

Bell, T.C., & Vahrenhold, J. (2018). CS Unplugged - How Is It Used, and Does It Work? *Adventures Between Lower Bounds and Higher Altitudes*.

Bocconi, S., Chioccariello, A., Dettori, G. et al. (2016). *Developing Computational Thinking in Compulsory Education - Implications for policy and practice*. EUR 28295 EN. Luxembourg: Publications Office of the European Union.

Bocconi, S., Chioccariello, A., Kampylis et al. (2022). *Reviewing Computational Thinking in Compulsory*

Education. JRC128347. Luxembourg: Publications Office of the European Union.

Curzon, P., & McOwan, P. (2017). *The Power of Computational Thinking.* London: World Scientific Publishing Europe Ltd.

Dasgupta, A., & Purzer, S. (2016). No Patterns in Pattern Recognition: A Systematic Literature Review. *IEEE Frontiers in Education Conference (FIE).* 1-3.

Denning, P., & Tedre, M. (2019). *Computational Thinking*. Cambridge, MA: MIT Press.

European Union. (2015*). Official Journal of the European Union*. Retrieved March, 20, 2022, from https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52015XG1215(02)&from=NL

European Union. (2018*). Official Journal of the European Union*. Retrieved March, 20, 2022, from https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32018H0604(01)&from=NL

Goldberg, D., Grunwald, D., Lewis, C. et al. (2013). Addressing 21st Century Skills by Embedding Computer Science in K-12 classes. In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13),* 637–638.

Guzdial, M. (2015). Learner-Centered Design of Computing Education: Research on Computing for Everyone. *Synthesis Lectures on Human-Centered Informatics*, *8*, 1-165.

Gretter, S., & Yadav, A. (2016). Computational Thinking and Media & Information Literacy: An Integrated Approach to Teaching Twenty-First Century Skills. *TechTrends, 60*.

Grover, S., & Pea, R. (2017). Computational Thinking: A Competency Whose Time Has Come.

Hemmendinger, D. (2010). A Plea for Modesty. *ACM Inroads*, *1*(2), 4–7.

Kafai, Y. B., & Proctor, C. (2022). A Revaluation of Computational Thinking in K–12 Education: Moving Toward Computational Literacies. *Educational Researcher, 51*(2), 146–151.

Lodi, M. (2020). Informatical Thinking. *Olympiads in Informatics: An International Journal*, Vilnius University, International Olympiad in Informatics, 14, 113-132.

Lowe, T., & Brophy, S.P. (2017). An Operationalized Model for Defining Computational Thinking. *2017 IEEE Frontiers in Education Conference (FIE)*, 1-8.

Lu, J., & Fletcher, G. (2009). Thinking about Computational Thinking. *ACM Sigcse Bulletin, 41,* 260-264.

Mishra, P., Yadav, A, Henriksen, D. et al. (2013). Rethinking Technology & Creativity in the 21st Century: Of Art & Algorithms. *TechTrends. 57,* 10-14.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. USA: Basis Books, Inc.

Rich, P. J., Egan, G., & Ellsworth, J. (2019). A Framework for Decomposition in Computational Thinking. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (*ITiCSE '19),* 416–421.

Samaey, G., Van Remortel, J., Bersini, H. et al. (2014). *Informaticawetenschappen in het leerplichtonderwijs*. Standpunten nr. 27. Brussel: KVAB.

Sands, P., Yadav, A. & Good, J. (2018). Computational Thinking in K-12: In-service Teacher Perceptions of Computational Thinking: Foundations and Research Highlights. *Computational Thinking in the STEM Disciplines: Foundations and Research Highlights,* 151-164.

Selby, C., & Woollard, J. (2013). *Computational Thinking: the Developing Definition.* University of Southampton (E-prints).

Selby, C. (2015). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (*WiPSCE '15),* 80–87.

Statter, D., & Armoni, M. (2016). Teaching Abstract Thinking in Introduction to Computer Science for 7th Graders. In *Proceedings of the 11thWorkshop in Primary and Secondary Computing Education (WiPSCE '16),* 80–83.

Thijs, A., Fisser, P., & Hoeven, M. van der. (2014). *21e eeuwse vaardigheden in het curriculum van het funderend onderwijs: een conceptueel kader*. Enschede: SLO.

Voogt, J., & Roblin, N. P. (2012). A Comparative Analysis of International Frameworks for 21st Century Competences: Implications for National Curriculum Policies. *Journal of Curriculum Studies*, *44*(3), 299-321.

Voogt, J., Erstad, O., Dede, C. et al. (2013). Challenges to learning and schooling in the digital networked world of the 21st century. *Journal of Computer Assisted Learning*, *29*.

Voogt, J., Fisser, P., Good, J. et al. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies, 20*.

Weintrop, D., Beheshti, E., Horn, M. et al. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology, 25,* 127–147.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33–35.

wyffels, F., Martens, B. & Lemmens, S. (2014). Starting from Scratch: Experimenting with Computer Science in Flemish Secondary Education. In *Proceedings of the 9th Workshop in Primary and Secondary Computing Education (WiPSCE '14),* 12–15.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational Thinking for Teacher Education. *Communications of the ACM, 60*(4), 55–62.

DOI: 10.34641/ctestem.2022.466

The TACTIDE EU STEM project : TeAching Computational Thinking with Digital dEvices

Marc Jansen[1,3], Nardie Fanchamps[2], Marcelo Milrad[3], Marcus Specht[4], Ali Hamidi[3]
[1] University of Applied Sciences Ruhr West, Germany
[2] Open University Nederland, The Netherlands
[3] Linnaeus University, Sweden
[4]Delft University of Technology, The Netherlands
marc.jansen@hs-ruhrwest.de, nardie.fanchamps@ou.nl, marcelo.milrad@lnu.se, m.m.specht@tudelft.nl, ali.hamidi@lnu.se

ABSTRACT
One major challenge the educational community is facing relates to how to effectively integrate computational thinking (CT) concepts and ideas into a particular school curriculum. Acquiring CT-skills by means of STEM offers rich opportunities within students´ education which may lead to learning gains. Previous research has shown that, to maximize the appeal and potential of CT learning environments, a precondition must be set first. The materials used must invite problem-based, inquiry-based and self-discovery learning, must be used without creating misconceptions and, above all, must give students the opportunity to acquire knowledge that can be directly transferred to everyday practice in an accessible manner. All the above puts demands on teachers who carry out learning and teaching in these environments. The EU funded TACTIDE project has tried to incorporate relevant curricular components into a coherent task, implementing assignments and challenges across different subjects and curricula of three different European countries. Based on the analysis of each national curricula, common topics have been identified and sub-scenarios have been developed. These sub-scenarios have been conceived to promote the integration between the topics mediated by CT. To achieve this objective, a greenhouse scenario has been conceptualized and designed towards teaching CT, by the use of microcontrollers such as the BBC micro:bit and the Calliope Mini, as an overarching STEM-topic. Using available sub-scenarios, a Moodle-course for teachers was developed for daily school activities to which other subjects in the core curriculum were interconnected in order to integrate CT skills and abilities. Scalability across different school levels and heterogeneous groups of learners, especially focusing prior knowledge, have been considered important design elements.

KEYWORDS
Computational thinking, teachers, curriculum, STEM, learning scenarios

1. INTRODUCTION
The emergence and application of new technologies in everyday life requires specific knowledge and skills. Through STEM-education these skills could be acquired, as STEM educational scenarios offer opportunities for an integrated subject matter approach combined with the use of digital tools. Previous research has shown that for maximizing the attractiveness and possibilities of novel learning environments, a precondition must be set first. The used materials should invite problem-based, inquiry-based and self-discovery learning, should be used without creating misconceptions and, above all, should give students the opportunity to acquire knowledge that can be transferred directly to everyday practices in an easily accessible way. The latest sets demands on teachers providing these environments. New forms of STEM education also more and more stress the importance of students´ digital literacy and the development of computational thinking skills.

Computational thinking is a way of approaching and solving problems using concepts from computer science and primarily involves the ability to reason, plan and solve problems (Wing, 2006). It refers to operationalised concepts such as parallel thinking, pattern recognition, completion, debugging, sequencing, and abstract reasoning that are needed to systematically approach a problem (Basawapatna, Koh, Repenning, Webb, & Marshall, 2011; Lee et al., 2011). Computational thinking involves the process in which problem definition, solution expression and implementation with evaluation recur in the process of programming (Yadav, Hong, & Stephenson, 2016), and can contribute to understanding and solving complex programming problems (Voskoglou & Buckley, 2012). Computational thinking encompasses in general two main directions: computational concepts and computational practices (Grover & Pea, 2017). CT concepts include: logic & logical thinking, algorithms & algorithmic thinking, patterns & pattern recognition, abstraction & generalization, evaluation, and automation. CT practices refers to: problem decomposition, creating computational artefacts, testing & debugging, iterative refinement, collaboration, and creativity.

A challenging discussion for promoting computational thinking education is how the acquisition of these skills can be integrated in the curriculum and how other subjects in the core of the curriculum are linked to this. The TACTIDE project has explored how to integrate relevant curricular components into a coherent educational activity by linking them to the creation of a greenhouse which integrates tasks and challenges from different subjects and across the curricula of three different European countries.

2. STATE OF THE ART
The application of programmable tangibles and artefacts is a playful integration of developing problem-solving skills and computational thinking. The application of robotics in STEM-contexts requires students to apply logical reasoning

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

in programming environments. It also demands systematic thinking, for the right choice of sensors and actuators, to program a robot that can anticipate the physical environment (Fanchamps, et al., 2019). Programmable robots harbour the potential to develop computational thinking skills because the visually observable output makes the results of programming interventions concrete and tangible (Catlin & Woollard, 2014; Sapounidis, Demetriadis, & Stamelos, 2015; Slangen, 2016). When users can test the effect of programming actions in authentic situations, they are better able to critically review and assess their programming actions (Slangen, Keulen, & Gravemeijer, 2011). Because programmable robots can be used to obtain immediate feedback on the consequences of code, they function as direct manipulation environments (DMEs) (Jonassen, 2006; Rekimoto, 2000). Direct manipulation environments (DMEs) involve users in constructing mental models of phenomena. Users are challenged to directly manipulate parameters and variables in the environment. Many DMEs reinforce the sense of operating with concrete objects. DMEs allow users to reason, predict, and hypothesise, analyse and test through active participation (Jonassen, 2006; Slangen et al., 2011). Robots are concrete and physical DMEs and can be controlled by programming using actuators and sensors (Jonassen, 2006; Rekimoto, 2000). They provide a potentially rich context for learning, understanding and practising programming and robotics concepts and for developing (general) problem-solving and computational thinking skills (CT).

The ability to anticipate changing environmental conditions by means of sensor observations and the computer program to be constructed, is a strategy to obtain an increased proficiency in computational requirements (Kim & Kim, 2003). To anticipate changes in the environment by means of sensor use requires a different program than performing programming tasks in an unchanging, predictable environment. By making use of sense-reason-act (SRA) programming, a programmed artefact or simulation of reality can react to changes in its surroundings. SRA-programming can be described as the process in which external, sensor-based observations (sense) are entered into a microprocessor, so that these observations can be compared with internal pre-set conditions (reason) which infers executing desired programming actions (act) (Fanchamps et al., 2019). The ability to anticipate changing conditions in the task design by means of sensor-based observations requires a different programming approach in comparison to linear solutions (Dragone et al., 2005). SRA-programming involves the functional understanding and application of complex programming concepts such as nested loops, conditionals and functions (Jansen, Kohnen-Vacs, Otero & Milrad, 2018; Slangen, 2016). Being able to respond to changes in the task design by means of SRA-programming can ensure that users' computational thinking ability develops at a higher level (Riedmiller & Gabel, 2007).

To teach computational thinking, teachers and designers should develop curricula to prepare and further enhance children's computational thinking competencies (Fanchamps et al., 2020). This by reinforcing the application of CT concepts and practices in the classroom. For learners, practicing and applying computational thinking concepts and approaches in contexts both within and outside of programming is an important prerequisite for acquiring skills that are required and applicable in other school subjects. For teachers this demands an adapted and tuned pedagogy to be able to integrate the cognitive and affective dimensions of deeper learning underlying computational thinking. For a thorough implementation of STEM in education and curriculum integration, the methodology of subject integration is proposed (Kelley & Knowles, 2016).

3. TEACHING CT ACROSS DIFFERENT SUBJECTS

To develop an integrated approach across different subjects and the implementation in the different partner countries, the curricula (grades 6-9) from Germany, The Netherlands, and Sweden have been compared to see which potential subjects could be integrated into a multidisciplinary course in CT concepts in order to enforce learning. To achieve this, an identification of the different courses was created after which for each country a tick was placed to show the presence of these subjects. In different stages of education, the amount and selected subjects may differ. Therefore, the age group of 12-15 year-olds was chosen as the starting point. In the analysis of the learning outcomes for this age group the common subjects and objectives between the different countries have been identified. This led to common objectives in mathematics, biology and physics. From the courses analyzed languages, creative, and social studies did not meet the requirements for creating a CT course. These courses could not be selected as they are not widely supported within all three countries. This left the STEM (Science Technology Engineering and Mathematics) subjects as the final choice.

Despite the macro-level strategy adopted by the different countries which are involved in this study in terms of how to integrate CT into the curriculum, there are numerous possibilities to put CT into practice. One possible opportunity is the integration of several subject matters within the context of designing and implementing a greenhouse scenario in connection to STEM. Indeed, our designed scenario bonds CT and STEM in a context where physical and digital tools are integrating and interacting with each other. Designing, creating, and experimenting in areas that are interesting for students are three crucial elements for such integration (Brennan & Resnick, 2012; Zerega et al, 2021). The students involved in these learning activities will be encouraged to use their creativity to design a portable indoors greenhouse. They will conceptualize their ideas and then create their designing thoughts in practice. Moreover, they will learn and share their knowledge during both the design, implementation, and experimentation phases. Moreover, students will learn some central environmental facts by observing, thinking, experimenting, and testing them during the various activities.

This approach could also provide the opportunity for critical thinking and developing problem-solving skills in two ways: 1) in the design and building phase of the greenhouse from 3D design and modeling to physical construction, and 2) in programming the BBC micro:bit that is mounted in the

greenhouse together with different sensors. For those who are new to programming, it provides an opportunity to take the first steps into the field and to learn basic concepts of a programming language by using visual block coding in an authentic setting. The focus is not on learning coding only but also on developing computational thinking skills and physical computing through coding the microcontrollers and sensors. In general, by using electrical components and simple electronics in combination with environmental science and programming, different aspects of STEM are explored during this proposed activity. The learning activities and learning modules for the integrated course have been collected in an online course which can be used by teachers to run the course in the actual classroom.

4. IMPLEMENTATION

A greenhouse, which can be operated via a programmable micro controller, has been used as a concrete elaboration of a DME. The application of CT skills is requested to program and control this greenhouse in a functional way. This greenhouse allows for science education in the form of growing plants and what the plants require for their growth, fostering our precondition that a corresponding learning scenario should invite problem-based, inquiry-based and self-discovery learning. Using a microcontroller, a program can be created to measure and provide the needs of these plants. Engineering would be covered by allowing the students to create their own model for the greenhouse. Finally, mathematics could be covered when, for example, the student needs to make calculations for how long it would take to cool down the greenhouse using the fan.

The goal of this project is to provide an appealing and challenging learning scenario in which participants build an automated portable greenhouse, in which programmable microcontrollers like BBC micro:bit and/or the Calliope are used in order to monitor and control important parameters such as temperature, humidity, soil moisture to ensure a suitable environment for plants. The design and construction of the greenhouse was carried out in 2 steps:

- Sketching the mini greenhouse by using a browser-based 3D design and modeling tool (Tinkercad app), as shown in figure 1. This will help to imagine the final product.
- Building the greenhouse using reusable straws, clear vinyl/PVC, and glue as shown in figure 2.

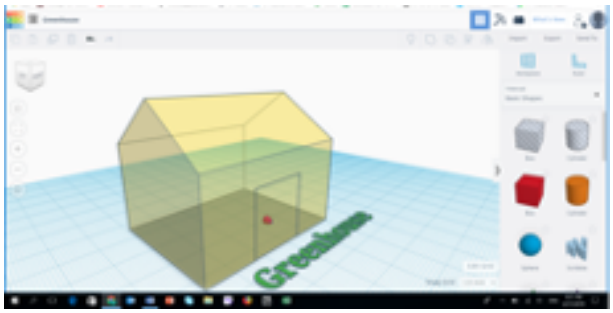


Figure 1. 3D Tinkercad design greenhouse.

Figure 2. Vinyl/PVC build greenhouse.

To control the above stated parameters physical computing devices and sensors are combined with visual programming. The coding part of the greenhouse project is done using the MakeCode editor (BBC micro:bit, 2019) as described in figure 3 depicted in the figure below:

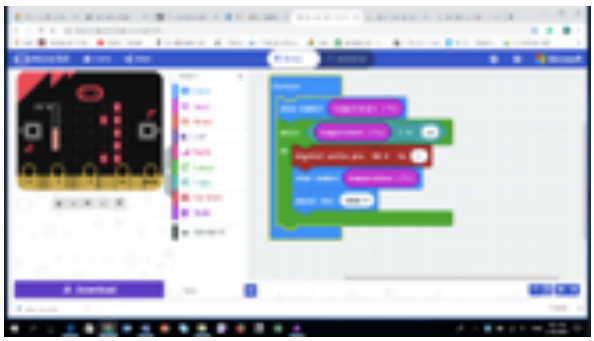


Figure 3. Coding BBC micro:bit.

After connecting a temperature/humidity sensor, a fan blade, a water pump, and a mini servo the following two scenarios could be realized:

Controlling temperature: According to the temperature measured the students can program the microcontroller to turn the fan when the temperature exceeds above a defined level. Similarly, the fan will be turned off automatically when the temperature comes below the defined level. This scenario includes specific tasks for reading sensor data from a thermometer, using conditionals to decide on different levels of temperature as also the use of loops for continuously measuring and controlling a fan until the temperature is in the target zone.

Controlling humidity: If the humidity comes either less than the defined limit or more than the optimum range, the fan, and the water pump will automatically turn on as well as rotating the mini servo to open the window on top of our greenhouse for better aeration.

In both scenarios, additional coding features and dimensions of computational thinking can be used such as creating functions to support working with patterns and problem decomposition.

5. SUMMARY AND OUTLOOK

In this paper we have presented the initial results of our efforts related to the conceptualization and development of a couple of educational scenarios that promote the integration between different school curriculum topics mediated by CT. The choice of content has been guided by

finding relevant subjects that emerged from the analysis of school curriculum in three European countries. Educational materials, including lessons plans, code examples, use of sensors and microcontrollers and video tutorials have been developed. Due to the Covid situation we have experimented since March 2020, the educational ideas and scenarios described above could not be validated with schools in these three countries.

Aspects that were planned to be assessed during the evaluation with students were related to conducting qualitative analyzes of the overall learning experience as well as the acceptance of this kind of learning scenarios in the schools.

It is important to emphasize that the TPACK framework (Mishra and Koehler, 2006) has been used to guide the development of the scenarios. Referring to the shortage in studies that focus on pedagogical aspects of teachers' CT development (Ottenbreit-Leftwich, et al., 2021), our work pays attention to aspects related to pedagogical content knowledge (PCK) as well as technological content knowledge (TCK) of the TPACK framework.

6. REFERENCES

Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011). Recognizing computational thinking patterns. Paper presented at the Proceedings of the 42nd ACM technical symposium on Computer science education.

Brennan, K. and Resnick, M., 2012, April. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).

Catlin, D., & Woollard, J. (2014). *Educational robots and computational thinking.* Paper presented at the Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education, Padova, Italy.

Dragone, M., O'Donoghue, R., Leonard, J. J., O'Hare, G., Duffy, B., Patrikalakis, A., & Leederkerken, J. (2005). Robot soccer anywhere: achieving persistent autonomous navigation, mapping, and object vision tracking in dynamic environments. Paper presented at the Opto-Ireland 2005: Photonic Engineering, Dublin, Ireland.

Fanchamps, N., Slangen, L., Hennissen, P., & Specht, M. (2019). The Influence of SRA Programming on Algorithmic Thinking and Self-Efficacy Using Lego Robotics in Two Types of Instruction. International *Journal of Technology and Design Education*, 1-20. doi:10.1007/s10798-019-09559-9

Fanchamps, N., Specht, M., Hennissen, P., & Slangen, L. (2020). The Effect of Teacher Interventions and SRA Robot Programming on the Development of Computational Thinking. Paper presented at the International Conference on Computational Thinking Education 2020, Hongkong.

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, 19.

Jansen, M., Kohen-Vacs, D., Otero, N., Milrad, M. (2018). A Complementary View for Better Understanding the Term Computational Thinking. In: Proceedings of the International Conference on Computational Thinking Education. 2018.

Jonassen, D. H. (2006). *Modeling with technology: Mindtools for conceptual change*. Upper Saddle River, New Jersey, USA: Pearson Merrill Prentice Hall

Kelley, T. R., & Knowles, J. G. (2016). A conceptual framework for integrated STEM education. *International Journal of STEM Education*, *3*(1), 1-11. doi.org/10.1186/s40594-016-0046-z

Kim, D.-H., & Kim, J.-H. (2003). A real-time limit-cycle navigation method for fast mobile robots and its application to robot soccer. *Robotics and Autonomous Systems*, 42(1), 17-30. doi:10.1016/S0921-8890(02)00311-1

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers' college record*, 108(6), 1017-1054.

Ottenbreit-Leftwich, A., Yadav, A., & Mouza, C. (2021). Preparing the Next Generation of Teachers: Revamping Teacher Education for the 21st Century. In *Computational Thinking in Education*, 151-171, Routledge.

Riedmiller, M., & Gabel, T. (2007). On experiences in a complex and competitive gaming domain: Reinforcement learning meets robocup. Paper presented at the 2007 IEEE Symposium on Computational Intelligence and Games.

Rekimoto, J. (2000). *Multiple-computer user interfaces: beyond the desktop direct manipulation environments.* Paper presented at the Conference on Human Factors in Computing Systems, The Hague, Netherlands.

Sapounidis, T., Demetriadis, S., & Stamelos, I. (2015). Evaluating children performance with graphical and tangible robot programming tools. *Personal and Ubiquitous Computing, 19*(1), 225-237. doi:10.1007/s00779-014-0774-3

Voskoglou, M. G., & Buckley, S. (2012). Problem solving and computational thinking in a learning environment. *Egyptian Computer Science Journal*, 36(4), 18.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, 49(3), 33-35. doi:10.1145/1118178.1118215

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational Thinking for All: Pedagogical Approaches to Embedding 21st Century Problem Solving in K-12 Classrooms. *TechTrends,* 60, 565-568. doi:10.1007/s11528-016-0087-7

Zerega, R., Hamidi. A., Tavajoh, S., & Milrad, M. (2021). A Co-design Approach for Developing Computational Thinking Skills in Connection to STEM Related Curriculum in Swedish Schools. In Proceedings of the *5th APSCE International Computational Thinking and STEM in Education Conference 2021*. Singapore: National Institute of Education, pp 144-147

DOI: 10.34641/ctestem.2022.467

How the Pre-service Teachers Associate Computational Thinking with Programming? A Case Study of an Introductory Programming Course in Teacher Education

Megumi IWATA[1*], Jari LARU[2*], Kati MÄKITALO[3*]
[1,2,3] Faculty of Education, University of Oulu, Finland
megumi.iwata@oulu.fi, jari.laru@oulu.fi, kati.makitalo@oulu.fi

ABSTRACT
There is a growing interest among educators to integrate computational thinking into basic education. Computational thinking is a complex concept and difficult to understand especially for those who have limited theoretical knowledge about this concept and no background in the computer science. Question arises, whether we reach the high-standard learning goals without comprehensive understanding of computer science. Therefore, there is a need to study computational thinking and how it should be introduced to pre-service teachers with little knowledge and experience in computer science and programming. This study aims to explore pre-service teachers' understanding of computational thinking in the context of an introductory programming course. We focus on to what extent the pre-service teachers recognize computational thinking during the course and how they associate their conceptual understanding of computational thinking with the concrete programming practices. We undertake in-depth analysis of five pre-service teachers who were novices in programming. The assignments and the survey after the course are analysed. The preliminary results show that sequencing from unplugged activities to computerized activities and project work helps the pre-service teachers recognized computational thinking. Understanding of the relationship between computational thinking and programming was diverse. Some explained that computational thinking helps understanding the code. This study provides insights of how computational thinking should be introduced along the way of learning programming.

KEYWORDS
Computational Thinking, Programming, Teacher Education, K-12 Education, Case Study

1. INTRODUCTION
Programming is a difficult subject for novices. Selby (2015) explains the learning difficulties of programming, which indicates the lack of ability to understand how a computational model works, to master reading, tracing and writing code, and to understand high-level concepts, such as design. Learning programming requires thinking and metacognitive skills, knowledge and information from multidisciplinary fields (e.g., Durak, Yilmaz, & Yilmaz, 2019; Selby, 2015; Li, 2016).

Wing (2006) states CT is a skillset for everyone. She defines that CT "is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Cuny, Snyder, & Wing, 2010, as cited in Wing, 2010, p.1). Later Aho (2012) redefines CT as the thought processes involved in formulating problems so that "their solutions can be represented as computational steps and algorithms" (p. 832).

While programming is a process to solve certain problems with computing, CT involves not only programming but also views which reflect benefits provided by computing for solving problems (Kukul & Karatas, 2019). Indeed Wing (2006) claims that CT is not programming but more like conceptualizing. Understanding such concept may be more challenging for those who have limited knowledge in computer science and experience in computer programming.

To advance CT education, developing teachers' ability is considered as a key factor (Kong, Abelson, & Lai, 2019). In teacher education, teachers not only obtain programming skills, but also understand CT and practice pedagogy to effectively foster CT. Despite the increasing interest in CT and programming among educators, the previous study found that many teachers had little understanding of CT skills and how they could develop CT skills in the practice (Sands, Yadav, & Good 2018). There is a need to develop teacher education to equip pre-service teachers with ability to think computationally to be able to integrate CT into education (Yadav, Gretter, Good, & McLean, 2017).

This study aims to explore how the pre-service teachers associate CT with programming in the context of an introductory programming course in teacher education. There has been less emphasis on how CT can help learning programming compared to how CT can be developed through programming. To achieve the aim, we set the following research questions: 1) To what extent do the pre-service teachers recognize CT in the introductory programming course? 2) In what ways do the pre-service teachers associate CT with programming?

With the first research question we investigate pre-service teachers' perceptions about CT in the introductory programming course. With the second research question, we aim to understand how the pre-service teachers relate conceptual CT with practical programming. The findings of the study can provide views on how CT should be introduced for programming novices along the way of learning programming.

2. PROGRAMMING AND CT
Previous literature review revealed the distinct emphasis on programming among the CT related literature (Saqr, Ng, Oyelere, & Tedre, 2021). Moreno-León, Román-González, and Robles (2018) claims that there are two main strategies to develop CT: unplugged activities and computerized activities. Unplugged activities include logic games, cards, puzzles to get to know computer science concepts.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Appropriate unplugged approach may help learning process of novices. For instance, Looi, How, Longkai, Seow, and Liu (2018) conclude that unplugged activities can possibly help understand key concepts of computing and develop CT.

Computerized activities, such as programming, expose students to CT using computer sciences concepts (Lye, & Koh, 2014). Dural and colleagues (2019) point out that thinking skills and knowledge in different fields required in the processes is considered as a basic strategy for developing CT and computer-based problem-solving process. Li (2016) suggests the close relationship between CT and programming course. CT should be the goal for the programming course because the focus is broader, problem-solving, and thinking skills not limited to programming language. The programming course can provide a practical carrier to the cultivation of CT ability because 1) programming is the best way to express CT, 2) programming course may include thinking methods of CT and 3) practices in programming course can provide opportunity to train CT (Li, 2016). Inquiry-based pedagogical approach includes problem solving and requires thinking skills, creativity and provides the platform for adapting theory to practise (Häkkinen, Järvelä, Mäkitalo-Siegl, Ahonen, Näykki, & Valtonen, 2017, Iwata, Laru, & Mäkitalo, 2020).

3. METHODS

This is a case study in which we explore pre-service teachers' experiences in an introductory programming course.

3.1. Participants

The participants are five pre-service teachers (Pre-service teacher A-E), who participated in the course (see Table 1). Pre-service teacher A, B and C were from primary teacher education program, and Pre-service teacher D and E were from subject teacher education program.

Table 1. Demographics of the Participants.

Pre-service teacher	Study in university	Teaching experience	Programming experience
A	1 year	None	None
B	1 year	None	None
C	3 years	1 year	None
D	4 years	1 year	None
E	1 year	1 year	None

3.2. Introductory Programming Course

The introductory course entitled *"programming in basic education"* is an optional course at the pre-service teacher education. This course corresponds to 5 ECTS [1] and is estimated as 133.5 hours of study including 20 hours of lectures, 30 hours of exercises, as well as self-study.

The main contents of the course included: 1) familiarizing oneself with collaborative problem-solving in the context of programming, 2) familiarizing oneself with the contents related to programming in the basic education curriculum, 3) practicing the basics of computational thinking, 4) getting to know different programming tools and environments for beginners, and 5) understanding the basics of automation with robotics tools. The tools used in the course were divided into five topics: 1) unplugged programming, 2) visual programming, 3) tinkering, 4) programming for games, and 5) robotics. In the spring 2021, 12 pre-service teachers participated in the course, which was organized as mixture of distance and face-to-face lessons (hybrid learning) because of the covid-19.

In the course, collaborative inquiry learning method was used as an approach to provide pre-service teachers experience on this kind of pedagogy. Assignments (group or individual) were given in each topic. Examples of the assignments were: *Create a coding project using the tool; Plan a small learning activity using the coding tool.* Pre-service teachers engaged in the project work, where they created learning materials for robotics programming activities with BBC micro:bits. The pre-service teachers were divided into two groups and created the learning materials consisted of multiple programming tasks. Pedagogically they were challenged by adding structure to adjust difficulties, examples and hints to help students proceed, and guiding questions to encourage reflection.

CT was introduced to the pre-service teachers in the beginning of the course and pointed out throughout the course along with learning of different topics. The programming skills by the National New Literacy Development Program *(Uudet Lukutaidot[2])* was used as a main framework in this course. Uudet lukutaidot is a joint program of the National Audiovisual Institute and the Ministry of Education in Finland. The framework describes programming related skills in three categories and nine subcategories. CT is one of the categories, which includes the four subcategories: logical thinking and information processing, problem solving and modelling, programming concepts and structure, and programming practices. This framework was chosen because it provides practical information for the teachers such as age-appropriate pedagogy and instructions. In addition, Brennan and Resnick's (2012) three dimensions of CT elements were explained by the teacher. The frameworks were provided as a foundation for the pre-service teachers so that they can recognize and practice CT by themselves while working on the course assignments and the project.

3.3. Data Collection and Analysis

The data for this study includes a survey filled by the pre-service teachers after the course and the assignments and the materials produced during the course. The survey included 16 questions about their experience during the course. CT was mentioned in the survey questions, such as "*CT was clearly part of this course*" and "*I understand how programming and CT are related".* The survey questions were answered with the five-point Likert scale followed by the further questions to ask the reasons of the choice.

The data was in Finnish which was later translated into English. The data was analysed inductively. First, the researchers read the data and familiarized themselves with the data. Then the researchers marked the parts of the data which were related to the research focus of the study. Those

[1] European Credit Transfer and Accumulation System (ECTS)

[2] https://uudetlukutaidot.fi/ohjelmointiosaaminen/

parts were categorized by themes. Processes of modifying the themes and dividing the data into themes were repeated.

4. RESULTS

4.1. Learning CT through Practicing Programming

Four out of five pre-service teachers agree that CT was clearly part of the course. However, the results indicate that pre-service teachers perceive CT differently in the course.

In the survey, two pre-service teachers indicate that CT was well visible in the topic of unplugged programming. Various activities and web resources for unplugged programming were presented to the pre-service teachers in the beginning of the course. The answers in the survey indicate that the structure, which starts with unplugged exercises and continues with visual programming, robotics and project work, can deepen understanding of CT. Two pre-service teachers' answers in the survey are as follows.

I think computational thinking was visible throughout the course. The course began with unplugged programming, which led to connecting with computational thinking. The exercises in the course were multifaced and developed computational thinking in different ways. For example, nice board games and apps (Scratch Jr + Scratch) led to solving problems piece by piece. (Pre-service teacher A)

I think computational thinking came up right at the beginning of the course when we program each other like a robot (unplugged). Immediately, such exercises provoked to think about computational thinking, which we then deepened through games, robotics, and project work. (Pre-service teacher D)

The assignment, where the pre-service teachers reflected on how CT was visible in the unplugged activities, shows the their understanding of CT and unplugged programming. Pre-service teacher A answered in the assignment as follows: "the student creates step-by-step instructions using simple commands and a repeat structure"; "the student recognizes the errors in the instructions and tries out solutions to correct them"; "the student develops precise and detailed instructions for using repeat and conditional structures".

The project work was explorative and ill-structured problem solving, where the pre-service teachers may apply CT. Pre-service teacher E expresses that she recognized CT in the problem-solving process during the project work.

Producing the content of project work required computational thinking; in particular, the content team had to think and come up with a wide range of problems and tasks, assess their difficulties, arrange these challenges to create meaningful entities, and consider possible different initial levels [of programming] to find meaningful things for everyone. When doing things, I did not notice, but after looking at it, I can see how the thought process has progressed and find the features of computational thinking there. (Pre-service teacher E)

The results indicate that pre-service teachers have various levels of understanding of CT through programming practices. The below quote shows that pre-service teacher C think that CT was not visible enough in the course.

In my opinion, the tasks and exercises of the course "forced" a different way of thinking and helped to develop computational thinking. However, there was little emphasis on thinking in the lessons, for example, and the perception of such thinking was not noticed until after the course. (Pre-service teacher C)

4.2. Relationship between Programming and CT

All five pre-service teachers show confident in understanding the relationship between programming and CT. Pre-service teacher D and E state that CT is behind practices in programming, such as problem solving, logical thinking and creative process. Pre-service teacher D explains that "It [CT] is about thinking, developing, problem solving like an IT expert or a computer would do. When you program, you get a certain kind of 'sense of control' about creating something new, more effective, and meaningful". Pre-service teacher E explains the connections between CT and programming practices as below.

Computational thinking is part of programming. It involves basic notions of programming, logical reasoning, and problem solving. Computational thinking is behind all programming activities, influencing action, thinking, and creation. Understanding a problem, finding a solution to it, and putting the solution into practice are all computational thinking and its outcome. Computational thinking thus serves as a kind of basis for all other programming activities. (Pre-service teacher E)

Two pre-service teachers mention that CT can be developed by programming. "Computational thinking can be taught through programming, for example, engaging in unplugged programming or programming with devices allows you to practice and develop computational thinking skills" (Pre-service teacher A).

Pre-service teacher B and C explain that CT helps to understand programming. With understanding of CT and programming concepts, the pre-service teachers may better understand programming practices including the meaning of the code. Pre-service teacher B wrote in the survey that "recognizing sequences and understanding the purpose of commands, these aspects combine computational thinking and programming" (Pre-service teacher B). Pre-service teacher C explained in the survey as follows.

Computational thinking allows general understanding of programming and makes it easier to understand how programming works. In particular, computational thinking is emphasized when looking at, for example, the operation and meaning of commands in programming. A logical mindset and the ability to perceive repetitive "rules" make it easier to understand how programming works. (Pre-service teacher C)

The assignments to read and remix others' code gave the pre-service teachers opportunity to practice using CT as a help to understand the code. In the assignment of visual programming, the pre-service teachers remixed existing Scratch projects. Using a Scratch game that adds a point when a character is clicked, pre-service teacher B remixed the game by adding a new character that reduces a point when it is clicked. In addition, she made two characters have

conversation. To do so, she needed to understand the code of the original character and make modifications in the code.

These are the preliminary results, which indicates that the pre-service teachers understood the meaning of CT and programming practices differently through this course. Further, more detailed perception on how pre-service teachers built their understanding about CT through different exercises will be presented at the conference.

5. DISCUSSION AND CONCLUSION

This study aims to explore how pre-service teachers associate CT with programming in the context of the introductory programming course. The course provided opportunities to practice CT through different programming assignments. Such opportunities can cultivate CT (Li, 2016) and encourage to think computationally, which is the first step for pre-service teachers to integrate CT (Yadav et al., 2017). We note three main findings from the preliminary results that can be used to improve the approaches to introducing CT along the way of learning programming. First, the pre-service teachers' perceptions on how CT relates with programming differ. One of the reasons is that relationship between CT and programming was not explained explicitly by the teacher but the pre-service teachers were expected to build understanding by themselves. Second, the results indicate that sequencing the learning topics from unplugged activities to computerized activities and project work, where all these learnt issues must be adapted, helps pre-service teachers to understand programming and acknowledge CT in relation to programming practices and a bigger picture of problem solving. Third, the results demonstrate that CT can help to understand programming. The pre-service teachers used CT to overcome the inability that causes the difficulties of learning programming, such as understanding how a computational model works, and mastering reading, tracing, and writing code, as Selby (2015) described.

As limitation of the study, we acknowledge that the number of the participants are small. We tried to understand the pre-service teachers who were novices in programming with multiple data sources. In future, interview methods and pre- and post-assessments of CT may give deeper insights of the pre-service teachers' understanding of CT. Pedagogical aspects should be explored more in the future studies, which addresses diverse ways to teach unplugged activities, games, and robotics in pre-service teacher education, to find out efficient approaches for learning CT and programming.

6. REFERENCES

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835.

Durak, H. Y., Yilmaz, F. G. K., & Yilmaz, R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology, 10*(2), 173-197.

Häkkinen, P., Järvelä, S., Mäkitalo-Siegl, K., Ahonen, A. K., Näykki, P., & Valtonen, T. (2017). Preparing teacher students for twenty-first-century learning practices: a framework for enhancing collaborative problem-solving and strategic learning skills. Teachers and Teaching: Theory and Practice, 23(1), 25-41.

Iwata, M. Laru, J., & Mäkitalo, K. (2020). Designing problem-based learning to develop computational thinking in the context of K-12 maker education. K. Kori K. & M. Laanpere (Eds.), proceedings of International Conference on Informatics in School: Situation, Evaluation and Perspectives, 103-106.

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior, 52*, 200-210.

Kong, S. C., Abelson, H., & Lai, M. (2019). Introduction to Computational Thinking Education. In Kong, S. C. & Abelson, H. (eds). *Computational Thinking Education*. Singapore: Springer.

Kukul, V., & Karatas, S. (2019). Computational thinking self-efficacy scale: Development, validity and reliability. *Informatics in Education, 18*(1), 151-164.

Li, Y. (2016, October 12-15). Teaching programming based on Computational Thinking. In *2016 IEEE Frontiers in Education Conference*, Erie, PA, 1-7. IEEE.

Looi, C. K., How, M. L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255-279.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior, 41*, 51-61.

Moreno-León, J., Román-González, M., & Robles, G. (2018, April). On computational thinking as a universal skill: A review of the latest research on this ability. In *2018 IEEE Global Engineering Education Conference* (EDUCON) 1684-1689. IEEE.

Sands P., Yadav A., Good J. (2018) Computational Thinking in K-12: In-service Teacher Perceptions of Computational Thinking. In Khine M. (eds) *Computational Thinking in the STEM Disciplines*. Springer, Cham.

Saqr, M., Ng, K., Oyelere, S. S., & Tedre, M. (2021). People, ideas, milestones: a scientometric study of computational thinking. *ACM Transactions on Computing Education (TOCE), 21*(3), 1-17.

Selby, C. C. (2015, November). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. In *Proceedings of the workshop in primary and secondary computing education,* 80-87.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2010). *Computational Thinking: What and Why?* The Link Magazine. Retrieved May 4, 2022, from https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In *Emerging research, practice, and policy on computational thinking,* 205-220. Springer, Cham.

DOI: 10.34641/ctestem.2022.468

Pedagogical Use of Scratch Coding for Co-Developing English Language "Locations and Directions" Building Blocks and Computational Thinking

Siu Cheung KONG *, Wai Ying KWOK
The Education University of Hong Kong, Hong Kong
sckong@eduhk.hk , waiyingk@eduhk.hk

ABSTRACT

This study designed and evaluated a pedagogical innovation which used Scratch coding to foster Grade 4 students to co-develop subject knowledge and computational thinking (CT) in English Language classrooms. A 280-minute teaching in four lessons was arranged for 205 students from ten selected Grade 4 classes in three primary schools at Hong Kong. The Scratch-based pedagogy "To Play and Learn, To Think and Navigate, To Code" – supplementary with four Scratch games and four Scratch activity worksheets – was trialed for engaging students in Scratch coding to explore, think about, apply and consolidate English Language building blocks for talking about locations and directions. The pre-post-test results provide statistically significant evidence that students can advance all three topic-specific knowledge points and all four target CT concepts after learning under the designed pedagogy. The questionnaire survey results reveal the impact of the designed pedagogy on students' growing awareness of the two target CT practices, and their growing confidence in coding. The focus group interview results reveal students' confirmation on their success in and satisfaction with the designed pedagogy for developing English Language knowledge and CT competency through coding. This study validates that the pedagogical innovation of learning through coding is potential to foster the co-development of subject knowledge and CT competency in Grade 4 English Language classrooms. Future directions of integrating coding activities into senior primary curriculum are discussed.

KEYWORDS

computational thinking, English Language, locations and directions, primary schools, Scratch coding

1. INTRODUCTION AND BACKGROUND OF STUDY

Computational thinking (CT) is growingly recognized to be a necessary competency for the success in the digitalized society (Grover & Pea, 2013; Shute, Sun, & Asbell-Clarke, 2017). CT is a thinking process, as defined in the seminal work by Wing (2006, p.33), of "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science". According to Brennan and Resnick (2012) and Rodríguez-Martínez, González-Calero, and Sáez-López (2020), the competency of CT cover three areas – CT concepts about the concepts central to programming such as sequence, conditionals, repetition; CT practices about the process of creating programming products such as iterative and incremental, abstracting and modularizing, testing and debugging; and CT perspectives about students' perspectives for problem-solving in the digitalized society. School education sectors around the world realize that it is important and necessary to integrate CT education in school curriculum to develop CT among students. With the popularity of the educational use of block-based programming environments such as Scratch, researchers such as Parsazadeh, Cheng, Wu, and Huang (2021) and Sarasa-Cabezuelo (2019) advocate the introduction of CT activities in school curriculum for students as young as at primary grades to co-develop subject knowledge and CT. This study addressed this curriculum advocacy to innovate a pedagogical design which engaged senior primary students in Scratch programming to co-develop "Locations and Directions" building blocks and CT competency in English Language classrooms.

The learning topic "Locations and Directions" is a major component in primary English Language curriculum (Taliancich-Klinger, Bedore, & Peña, 2018; Williams, 2020). There are three knowledge points in the learning scope of this topic – 1) the vocabulary of places in the street, such as "bakery", "bank", "footbridge", etc.; 2) the prepositions / prepositional phrases for describing location of a place, such as "opposite", "behind", "next to", etc.; and 3) the prepositions / prepositional phrases for giving directions to a place, "turn right", "go straight", "walk across", etc. (Chen & Lee, 2018; Taliancich-Klinger et al., 2018). These are important building blocks for generating the information and procedural texts for communication to indicate directions and give instructions. To cultivate young children with these English Language building blocks, researchers such as Taliancich-Klinger et al. (2018) and Williams (2020) suggest that subject classrooms should provide students with ample opportunities to meaningfully learn and apply these grammatical constructs in authentic communication contexts. With the growth of e-Learning in recent decades, more and more English Language classrooms integrate the use of digital educational games in the subject learning and teaching process for motivating students to explore, construct, and consolidate building blocks for English Language communication in an interesting and interactive manner (Chen & Lee, 2018; Wu, 2018).

In primary education, there is a popularity of using block-based programming environment Scratch in subject classrooms (Sarasa-Cabezuelo, 2019; Rodríguez-Martínez et al., 2020). Scratch has an intuitive interface-design of which children can make simple actions to drag, drop, and combine code blocks for an easy creation of programs and an immediate observation of programming outcomes (Parsazadeh et al., 2021; Sarasa-Cabezuelo, 2019).

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Regarding the use of Scratch programming environment for integrating CT education into subject curriculum, the frameworks proposed by Brennan and Resnick (2012) and Grover et al. (2017) are widely referred. Such curriculum integration has a rationale that the coding products serve as computational manipulatives which conceptually align with the traditional notion of educational manipulatives (Parsazadeh et al., 2021; Rodríguez-Martínez et al., 2020).

In the English Language subject curriculum for Primary 4 in Hong Kong school education, the knowledge taught in the topic "Locations and Directions" is procedural by nature (i.e. information and procedural texts for communication to indicate directions and give instructions). Such nature is very similar to the one of coding (as the essence of coding is to generate procedural commends for operating solutions on the computing devices). The topic "Locations and Directions" gives a natural point of introducing coding education elements into the curriculum delivery in the subject English Language. This topic is therefore selected as a point of pedagogical trial for learning English Language through coding in this study – for the design of subject-specific pedagogical innovations for Primary 4 students to learn information and procedural texts for communication using authentic scenario through computational tools.

Researchers suggest three main criteria for designing pedagogies which effectively integrate CT education in subject curriculum. The first criterion, according to Sarasa-Cabezuelo (2019) and Rodríguez-Martínez et al. (2020), is the provision of enough chances for students to work on the selected coding products to construct subject knowledge and stimulate their interest in coding. The second criterion, according to Parsazadeh et al. (2021) and Rodríguez-Martínez et al. (2020), is the provision of enough chances for students to apply subject knowledge in thinking about programming solutions for solving problems in subject-specific contexts. The third criterion, according to Parsazadeh et al. (2021) and Sarasa-Cabezuelo (2019), is the provision of enough chances for students to consolidate subject knowledge in generating coding products for solving subject-specific problems.

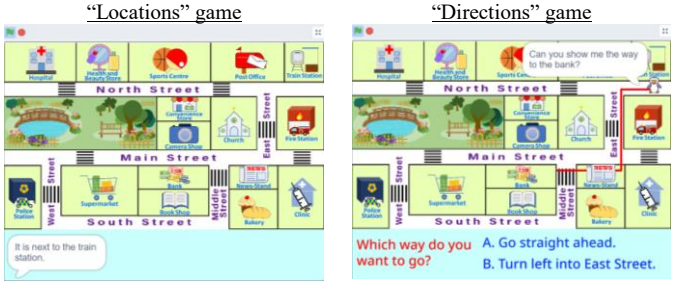
2. THE STUDY: RESEARCH DESIGN AND METHODOLOGY

This study pioneered a pedagogical design which integrated block-based programming activities in subject classrooms for supporting students to co-develop both subject knowledge and CT competency. The pedagogical innovation engaged students in Scratch coding for developing three types of building blocks for communication to indicate directions and give instructions (i.e. the vocabulary of places in the street as well as the prepositions and prepositional phrases for describing location of and giving directions to a place); and at the same time the competency of four CT concepts ("sequences", "conditionals", "repetition", and "operators") and two CT practices ("iterative and incremental" and "testing and debugging") – as in the CT frameworks by Brennan and Resnick (2012) and Rodríguez-Martínez et al. (2020).

A three-step pedagogy "To Play and Learn, To Think and Navigate, To Code" – supplementary with a Scratch programming environment with four Scratch games and four Scratch activity worksheets – was designed for engaging students in "playing" the Scratch games featured with Scratch-based interactive maps for "learning" the building blocks for talking about locations and directions when using maps; "thinking" of and "navigating" the movement actions of the sprites on the Scratch-based interactive maps to move from one place to another place; and "coding" in a Scratch template to create a Scratch animation with their own relocation of places on the interactive map, their own selection of sprites to be appeared on the interactive map, and their own design of routing for the sprites to walk from one place to another place (see Figure 1).

(a) Scratch games for the "To Play and Learn" step

"Locations" game | "Directions" game



(b) Scratch games for the "To Think and Navigate" step

"Treasure hunt" game | "Can you show me the way?" game

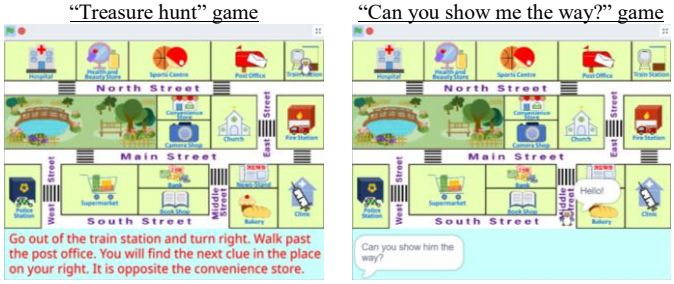


Figure 1. Scratch Games for the Designed Pedagogy.

A total of 205 students – from ten Grade 4 classes in three primary schools at Hong Kong – participated in this study (see Table 1). The English Language teachers of the ten participating classes made the class-specific trial of the designed pedagogy.

Table 1. Profile of Students Participated in This Study.

	School A	School B	School C
No. of students	100	59	46
No. of classes	5	2	3
Boys : Girls	52:48	24:35	27:19
Mean age (years)	8.91	8.96	8.96

The participating teachers completed a four-hour training workshop for their preparation before the trial teaching – covering the rationale of learning through coding in local

English Language curriculum; the implementation of the "To Play and Learn, To Think and Navigate, To Code" pedagogy in English Language classroom; and the integration of the four Scratch games and the four Scratch activity worksheets into topic-specific lessons. Two research questions were focused in this study: (1) What did the students achieve in developing English Language building blocks and CT under the pedagogical innovation? (2) How did the students perceive the pedagogical innovation for developing CT in English Language classrooms? This study adopted three methods for evaluating the designed pedagogy.

The first method was pre-post-tests at the beginning and the end of the designed pedagogy. It aimed to investigate students' achievement in English Language learning and CT development. The test papers contained eight questions: one question on testing the vocabulary of places in the street; two on the prepositions and prepositional phrases for describing location of a place; one on prepositions and prepositional phrases for giving directions to a place; and four on CT concepts including "operator", "repetition", "conditionals" and "sequence". Students' pre-test and post-test scores were statistically compared with the assistance of SPSS software. The Cronbach's alpha reliability coefficients for the pre-test and post-test are 0.81 and 0.82 respectively.

The second method was pre-post-surveys at the beginning and the end of the designed pedagogy. It aimed to investigate students' perception of developing CT in English Language classrooms. The questionnaire contained five 5-point Likert scale questions: three questions on the building of awareness, interest and confidence in coding, and two on the development of CT practices of "iterative and incremental" and "testing and debugging". The mean rating for each question and the corresponding standard deviation were then calculated. The Cronbach's alpha reliability coefficients for the pre-survey and post-survey are 0.86 and 0.84 respectively.

The third method was focus group interviews at the end of the designed pedagogy. It aimed to investigate students' perception of the designed pedagogy. There were 11 students randomly selected from the three participating schools for three focus groups, with each consisting of three to four students. The student respondents were asked to describe the helpfulness of the designed pedagogy in their development of English Language knowledge and CT competency, express how they enjoyed and were satisfied with the pedagogy for English Language learning through coding, and discuss the challenges in and suggestions on English Language lessons integrated with coding activities. All the interview content was transcribed and systematically summarized.

3. Results and Discussion

3.1. Students' Achievement in Developing Topic-specific Building Blocks and CT under the Designed Pedagogy

The pre-post-tests results show that the designed pedagogy effectively supported students to develop building blocks for communication to indicate directions and give instructions (see Table 2). There is a statistically significant increase in students' post-test scores for the question items on all three types topic-specific building blocks. This shows that students after the designed pedagogy had a noticeable gain in the knowledge about the vocabulary of places in the street as well as the prepositions and prepositional phrases for describing location of and giving directions to a place.

Table 2. Students' Achievement in Developing Topic-specific Building Blocks under the Designed Pedagogy.

Question items			***Pre-test***	***Post-test***	
Topic-specific building blocks	***No. of items***	***Max. scores***	***Mean (SD)***	***Mean (SD)***	***t-test***
A. Vocabulary - Places in the street	1	6	4.10 (1.80)	5.07 (1.31)	7.35***
B. Prepositions / Prepositional phrases - Describe location of a place	2	6	3.07 (1.80)	4.81 (1.54)	12.14***
C. Prepositions / Prepositional phrases - Give directions to a place	1	8	2.19 (2.03)	5.80 (2.08)	20.80***
Total	**4**	**20**	**9.36 (4.39)**	**15.69 (3.94)**	**19.54*** **

****p < 0.001*

The pre-post-tests also found that the designed pedagogy effectively supported students to develop CT concepts. There is a statistically significant increase in students' post-test scores for the question items on all four target CT concepts (see Table 3). This shows that students after the designed pedagogy noticeably enhanced their concepts used in coding.

Table 3. Students' Achievement in Developing Computational Thinking under the Designed Pedagogy.

Question items			***Pre-test***	***Post-test***	
CT concepts	***No. of items***	***Max. scores***	***Mean (SD)***	***Mean (SD)***	***t-test***
A. Operator	1	1	0.28 (0.45)	0.62 (0.49)	8.46***
B. Repetition	1	1	0.35 (0.48)	0.46 (0.50)	2.44*
C. Conditional	1	1	0.23 (0.42)	0.46 (0.50)	5.26***
D. Sequence	1	1	0.24 (0.43)	0.41 (0.49)	3.57***
Total	**4**	**4**	**1.11 (1.02)**	**1.96 (1.19)**	**8.29*** **

**p < 0.05 ***p < 0.001*

3.2. Students' Perception of the Designed Pedagogy for Learning English Language through Coding

The students positively perceived the designed pedagogy for developing CT in English Language classrooms (see Table 4). The students had a statistically significant increase in the level of agreement with the importance of the step-by-step development of a program and the operability-testing of the program after the trial teaching. This show that the students after the designed pedagogy noticeably enhanced their CT practice of "iterative and incremental" and "testing and debugging". The students after the trial teaching also had a statistically significant

growth of their confidence in programming and their understanding of the importance of programming.

Table 4. Students' Perception of the Designed Pedagogy for Developing Computational Thinking in English Language Classrooms.

Items	*Pre-survey* *Mean* # *(SD)*	*Post-survey* *Mean* # *(SD)*	*t-test*
I think it is important to develop a program step by step.	3.61 (1.36)	4.04 (1.13)	3.65***
I think it is important to test the program to make sure it works.	3.55 (1.35)	3.95 (1.23)	3.39***
I am interested in learning programming.	3.40 (1.37)	3.58 (1.28)	1.58
I think that programming is important in our daily lives.	3.30 (1.26)	3.51 (1.17)	1.97*
I am confident that I can write a simple program.	3.24 (1.37)	3.68 (1.21)	3.77***

#Note: 1=strongly disagree, 2=disagree, 3=neutral; 4=agree; 5=strongly agree.

**p < 0.05 ***p < 0.001*

The results of the focus group interviews, echoing with those of pre-post-tests and surveys, further confirmed students' positive perception of the designed pedagogy for developing CT in English Language classrooms (see Table 5).

Table 5. Feedback from Students' Focus Group Interviews on the Perception of the Designed Pedagogy.

What and how the designed pedagogy helped for the learning trial?
-The designed pedagogy enabled students to effectively learn about the vocabularies, prepositions and prepositional phrases for describing locations and giving directions when using maps. -The designed pedagogy enabled students to effectively develop Scratch coding knowledge and CT competency, in particular the components of "Conditional", "Repetition", and "Testing and Debugging".
How enjoyable and satisfactory was the designed pedagogy?
-The students liked the designed pedagogy for developing both subject knowledge and CT competency in a happy and interesting manner, as coding activities were funny and appealing. -One-third of students made additional efforts to play and re-create the Scratch game in the designed pedagogy after class time for exploring more the building blocks used to describe locations and give directions.
What are the challenges in and suggestions on this pedagogy?
-One-fourth of students at the beginning found the coding activities challenging, as Scratch programming environment was new to them and thus hard to recall the names and types of Scratch coding blocks for use. -One-third of students suggested an enrichment of the Scratch game scenario for learning both the topics of "Locations and Directions" and "Food and Drinks" at the same time specific for Grade 4 curriculum.

The students asserted the help of the designed pedagogy in the development of English Language building blocks and CT. Nearly three quarters of the student respondents confirmed that the designed pedagogy enabled them to effectively learn about the vocabularies of names of different places; and the ways to describe locations and directions in map-usage. A student respondent illustrated that after confirming the correct route he deliberately manipulated the sprite to go in the wrong and/or opposite directions, for a more interesting exploration of the locations appeared on the map. More than a third of the student respondents confirmed that the designed pedagogy enabled them to effectively learn about coding with Scratch to create games. Two student respondents pointed out that they have particular deep impression of CT concepts "Conditional" and "Repetition". The other two student respondents further explicated that the coding experience in the designed pedagogy fostered their awareness to test and debug coding products through the trial-and-error process.

The students expressed their high level of enjoyment in and satisfaction with the designed pedagogy for English Language learning through coding. All student respondents indicated that they had a happy and interesting experience in learning the target subject topic; and were greatly attentive during the coding activities which were funny and appealing. Around three quarters of the student respondents expressed that they liked the learning activities in the designed pedagogy. They considered the design of coding activities was easy to follow; and they gained a great sense of achievement when they successfully completed the coding activities. Nearly half of the student respondents indicated that they played the Scratch game after class time for learning exploration. Four of the students stated that they successfully altered the codes in the Scratch game after class time for creating new sprites and changing locations of the buildings on the map. One of them further noted that he had introduced his re-created Scratch game to his family members and friends for game-playing.

The students indicated challenges in and made suggestions on English Language lessons integrated with coding activities. As for the challenges in the designed pedagogy, around a third of the student respondents indicated that it was the first time for them to use Scratch; and at the beginning they found the coding activities were not easy as they were unfamiliar with Scratch programming environment. It took time for them to familiarize with the names and functions of the coding blocks. These student respondents therefore expected for pre-training of Scratch coding before the trial lessons. As for the recommendations on the designed pedagogy, more than half of the student respondents suggested that the designed pedagogy can be extended to the teaching of other topics in the English Language subject curriculum at the same learning grade (i.e. Grade 4). Five student respondents detailed their suggestion on a possible enrichment of the learning scenario of the designed pedagogy by combining the existing topic of "Locations and Directions" with the other topic "Food and Drinks" in the Grade 4 English Language subject curriculum. These students expected that building on the existing Scratch map, a zoom-in view of the places "Restaurant" and "Supermarket" can be added for game-extension: the sprites enter these places and make dialogs for making orders of food and drinks services in the restaurant and inquiring locations and prices of food and drinks products in the supermarket. Students in the enriched learning scenario can apply and consolidate knowledge about, for example, vocabularies of various types of food and drinks and sentence-making for making catering orders in restaurants and making general inquires in supermarkets.

4. CONCLUSION AND FUTURE WORK

This study pioneered a Scratch-based pedagogical design for Grade 4 students at Hong Kong for their co-development of English Language knowledge and CT competency in subject classrooms. The designed pedagogy focused on the development of building blocks in the English Language subject topic "Locations and Directions"; as well as four CT concepts and two CT practices. A four-lesson trial teaching was arranged for 205 students from ten selected Grade 4 classes in three Hong Kong primary schools to implement the pedagogy "To Play and Learn, To Think and Navigate, To Code", supplementary with four Scratch apps and four Scratch activity worksheets. From the pre-post-tests, the designed pedagogy is confirmed to be effective to support students to significantly enhance their knowledge about the vocabulary, prepositions and prepositional phrases for describing location of and giving directions to a place; as well as CT concepts of "operator", "repetition", "conditional" and "sequence". From the questionnaire surveys, the designed pedagogy is confirmed to be effective to support students to significantly enhance their CT practices of "iterative and incremental" and "testing and debugging". From the focus group interviews, the design and effectiveness of the Scratch-based pedagogy are confirmed to be well-received by the students for English Language learning and CT development through coding.

This study collected concrete suggestions on the potential expansion of topic coverage in the design of learning scenarios for Scratch-based interactive map games. Future work will consider these promising ideas for enhancing the design of games and worksheets of this Scratch-based pedagogical innovation to enrich students' experience in learning information and procedural texts for communication in senior primary English Language curriculum. Future work will also try to arrange a control group in the research design, in order to address the limitation in this study that no control group was involved in the evaluation of the effectiveness of the designed pedagogy. For providing a more accurate picture on the feasibility of smooth integration of coding elements into English Language classrooms, concerns will also be placed on the issues about the readiness of subject teachers, the sufficiency of class hours, and the solutions for cross-disciplinary barriers when preparing for the actual teaching arrangements of future trials of the pedagogical approach of learning English Language through coding.

5. REFERENCES

Bell, H., & Ainsworth, S. (2021). Difficulties assessing knowledge of grammatical terminology: Implications for teacher education and teaching. *Language Awareness*, *30*(2), 97-113.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *2012 Annual Meeting of the American Educational Research Association (AERA'12)*, Canada.

Chen, Z.-H., & Lee, S.-Y. (2018). Application-driven educational game to assist young children in learning English vocabulary. *Educational Technology & Society*, *21*(1), 70-81.

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

Parsazadeh, N., Cheng, P.-Y., Wu, T.-T., & Huang, Y.-M. (2021). Integrating computational thinking concept into digital storytelling to improve learners' motivation and performance. *Journal of Educational Computing Research*, *59*(3), 470-495.

Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: An experiment with sixth-grade students. *Interactive Learning Environments*, *28*(3), 316-327.

Sarasa-Cabezuelo, A. (2019). Use of Scratch for the teaching of second languages. *International Journal of Emerging Technologies in Learning*, *14*(21), 80-95.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142-158.

Taliancich-Klinger, C. L., Bedore, L. M., & Peña, E. D. (2018). Preposition accuracy on a sentence repetition task in school age Spanish–English bilinguals. *Journal of Child Language*, *45*(1), 97-119.

Williams, M. (2020). Fifth graders' use of gesture and models when translanguaging during a content and language integrated science class in Hong Kong. *International Journal of Bilingual Education and Bilingualism*, 1-20.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Wu, T.-T. (2018). Improving the effectiveness of English vocabulary review by integrating ARCS with mobile game-based learning. *Journal of Computer Assisted Learning*, *34*(3), 315-323.

DOI: 10.34641/ctestem.2022.469

Computational Thinking Dashboard: For learners in Jupyter notebooks

Bhoomika Agarwal

TU Delft, Netherlands

bhoomika10@gmail.com

ABSTRACT

Computational Thinking (CT) - the process of thinking like a programmer or computer scientist - is a skill that that has the potential to transform the way students learn at educational institutions in different domains and different grade levels. With the increasing integration of CT in classrooms, there is a growing need for CT assessment tools to evaluate the acquisition of CT skills. This research develops a framework for CT assessment that detects user micro-interactions in a university-level self-paced Python beginners course integrated into Jupyter notebooks. The users can improve their learning with the help of feedback via CT dashboards as part of this framework. A user evaluation study was conducted which showed that this framework can be used to improve the acquisition of CT skills via programming. The main contributions of this framework are the mapping between CT skills and user micro-interactions and development of the CT dashboards to help the user self-regulate their learning of programming. The framework developed can be easily integrated into any course that teaches Python programming using Jupyter notebooks and is yet to be extended to other programming courses.

KEYWORDS

computational thinking, programming education, python, jupyter notebook

1. INTRODUCTION

With the rapid integration of computers and technology into our daily lives in the 21st century, we are amid the technology revolution. While it is not yet necessary to learn to program or code, most of us use computers on a daily basis. We need to learn to think like them to get the best of this revolution. This process of thinking like a programmer or a computer scientist is called Computational Thinking(CT). As per the National Research Council of the National Academy of Sciences in the United States of America, CT is a skill that everyone should acquire, not just programmers (National Research Council et al., 2010) (National Research Council et al., 2011).

Computational Thinking is a concept that lacks an agreed-upon definition (Tang et al., 2020)(Brennan & Resnick, 2012)(Barr & Stephenson, 2011)(National Research Council et al., 2011). Brennan & Resnick, (2012) defined CT with respect to design-based learning activities in Scratch - a block-based programming language - in terms of three dimensions: computational concepts, computational practices, and computational perspectives. The concept of Computational thinking was brought to the limelight in 2006 when Wing, (2006) suggested that thinking computationally was a fundamental skill for everyone, not just computer scientists, and argued for the importance of integrating computational ideas into other subjects in school. Computational thinking has been shown to be a valuable skill for other domains and disciplines such as mathematics and science. Multiple studies have looked at CT skills as a transfer skill and how it can be applied in other domains (Weintrop et al., 2016)(Pei et al., 2018)(Leonard et al., 2018)(Jaipal-Jamani & Angeli, 2017).

A majority of the cross-disciplinary research makes use of visual and block-based programming languages. This graphical representation of code makes it easier to learn the basics of programming, especially for K-12 students and makes it suitable for integrating it into curricula in other domains. On the downside, the functionality of block-based programming language is limited by the available blocks and they do not offer the flexibility that text-based programming languages provide. Tang et al., (2020) show that a majority of studies related to CT are focused on elementary and middle school grade levels and emphasize on the need for more studies for high school and college students so that the complete development trajectory for CT skills in students can be mapped.

While Computational thinking (CT) is being integrated into curricula rapidly, there is a need for methods to assess and evaluate learning of CT concepts (Hadad & Lawless, 2015)(Tang et al., 2020). The lack of an agreed-upon definition of CT, lack of assessment mechanisms for CT and lack of usage of CT in classrooms are the major roadblocks in the integration of CT into curricula(Lyon & Magana, 2020). Owing to the advantages of a combination of assessments, my research will use a combination of a portfolio assessment and an adapted version of the survey scale developed by Kılıç, Gökoğlu, and Öztürk, (2021) to assess the programming-oriented CT skills of undergraduate students. By using this combination, the attitudes of the users towards CT skills can be measured using the scale and a holistic view of the users' CT skills can be gained through the portfolio assessment.

LA dashboards are learning tools that can help learners and teachers harness the power of LA use it to improve their learning (Jivet et al., 2020). Schwendimann et al., (2016) define LA dashboards as "a single display that aggregates different indicators about learner(s), learning process(es) and/or learning context(s) into one or multiple visualizations". By making the learner aware of their

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

progress and triggering self-reflection, LA dashboards can help users regulate their own learning (Jivet et al., 2017). Online learning provides flexibility and accessibility to students through increased learning opportunities, access to learning resources and opportunities for collaboration (Broadbent & Poon, 2015). The downside of online learning is that its success relies heavily on independent learning and the students' autonomous engagement in the course (Broadbent & Poon, 2015). SRL strategies can help learners to gain and retain knowledge methodically and systematically (Broadbent & Poon, 2015). My research aims to regulate the learning process and direct the student learning process with knowledge and feedback in the form of an SRL dashboard that includes the sense-making factors and support for action for SRL.

2. DESIGN AND IMPLEMENTATION

2.1. The Python Programming Course

The CT assessment framework is integrated into the Python basic programming. This course is a self-paced course to teach Python programming without any pre-requisite knowledge to university students. It is comprised of 4 modules – Variables, Control flow, Code Organization, Basic plotting. The course is based on Jupyter notebooks to allow for active leaning and experimentation and uses nbgrader for releasing the exercises. The code in these notebooks is runnable, producing output, and can be modified by the student, to learn all the details and study the effects of changes and variations.

2.2. Adapted definition of CT

For this research, an adapted definition of Computational Thinking(CT) that combines those by Brennan and Resnick, (2012) and Yeni and Hermans, (2019) is used. Brennan and Resnick, (2012) define CT for Scratch with 3 key dimensions: "computational concepts (the concepts designers employ as they program), computational practices (the practices designers develop as they program), and computational perspectives (the perspectives designers form about the world around them and about themselves)". Yeni and Hermans, (2019) adapt this definition to Python by modifying the CT concepts list to one that is better suited to Python. Visualization, also referred to as 'Simulation' or 'Modelling' is an important CT concept that is missing in the above definition(Hambrusch et al., 2009)(Weintrop et al., 2016)(International Society for Technology in Education & Computer Science Teachers Association, 2011)(Yuen & Robbins, 2014). Thereby my research adds 'Visualizations' to the list of CT concepts proposed by Yeni and Hermans, (2019). Thereby, the revised list of CT concepts used in my research is: data structures, operators, conditionals, sequences, loops, visualization.

Brennan and Resnick, (2012) identify 4 CT practices as part of their CT definition in the form of micro-interactions: being incremental and iterative, testing and debugging, reusing and remixing, abstracting and modularizing. My research uses these 4 CT practices and detects them through the user's micro-interactions.

2.3. CT Concepts mapping

This research uses 7 CT concepts and maps them to the 4 learning modules in the Python basic programming course. This mapping is used for the design of the module-wise dashboards. The CT concepts are mapped to the learning modules as shown in Table 1.

Table 1. CT Concepts Mapping

Module	CT concepts
Variables	Data, Operators
Control Flow	Loops, Conditionals
Code Organization	Sequences, Functionals
Basic Plotting	Visualization

2.4. Micro-interactions

Micro-interactions are the small-scale interactions that the user does with a platform such as keypresses, mouse button presses, copy and paste, etc. They can be useful to track the user behavior in real-time and provide feedback about their learning process. Micro-interactions can be aggregated and grouped to provide learning indicators that can help users with self-regulation of their learning process Matcha et al., (2020). This research collects micro-interaction data and processes them to form indicators of CT skills. There are two sources of the micro-interaction data - LogUI and notebook metadata.

Table 2. Micro-interactions mapping.

Micro-interaction	Action	Source	CT practice
focusin + focusout	Time spent on a cell	LogUI	BII
keystrokes	Additions	LogUI	BII
Cell run count	-	Notebook metadata	TD
Errors in output	Errors	Notebook metadata	TD
copy	Copy	LogUI	RR
paste	Paste	LogUI	RR
Add functions	-	Notebook metadata	AM
Module import	-	Notebook metadata	AM

LogUI is a framework-agnostic client-side JavaScript library developed by Maxwell and Hauff, (2021) for logging user interactions on webpages. Jupyter notebook stores its cells as an array of JavaScript Object Notation(JSON) objects. This contains metadata about the number of cell runs, errors, cell source and more. This research uses LogUI integrated into Jupyter notebooks together with Jupyter notebook metadata to detect micro-interactions such as the time spent on a cell, copy and paste. These micro-interactions are then aggregated to learning paramaters as shown in Table 2. The four CT practices defined by Brennan and Resnick, (2012) are: Being Incremental and Iterative(BII), Testing and Debugging(TD), Reusing and Remixing(RR), Abstracting and Modularizing(AM). For example, the number of copy-paste actions can indicate reuse of code in learning. These micro-interactions are then used as input for a global SRL dashboard.

2.5. CT Dashboards

Dashboards are tools that support both students and teachers by helping them make sense of the learning analytics data such that it can be used to improve the learning process (Jivet et al., 2020). Dashboards can be used to trigger learners to think about the effort invested in learning and the subsequent outcomes of these activities (Jivet, 2016). Dashboards are used in this research to provide feedback to students per module and about how the micro-interaction data can be used to improve the learning process. In this way, the students can regulate their learning process themselves. As the course is in Jupyter notebooks, the dashboards are also integrated into Jupyter notebooks so that the user does not have to use any additional tools or environments.

2.5.1. CT concepts dashboard

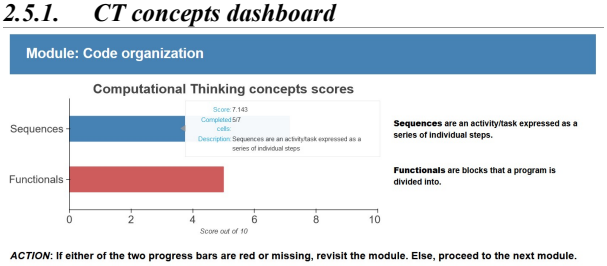


Figure 1. Module-wise CT Concepts dashboard

The user is provided feedback for self-regulated learning via Computational Thinking(CT) concepts dashboards per module. This dashboard uses metadata tags for the cells and checks the completion using certain conditions. Additionally, the user is provided with actionable suggestion for iterative self-regulated learning, as shown in Figure 1. Figure 1 shows the dashboard for the module 'Code Organization', covering 2 CT concepts – Sequences and Functionals. The progress of each concept is shown by a progress bar. This progress is computed by the ratio of the number of cells tagged with a concept that have been completed by the user against the ratio of the total cells tagged with a concept, scaled to a CT concept score of 1-10. The color of the progress bar is red if the progress is less than 60% of this ratio, as can be seen for the concept Functionals. The user is advised to revisit the module if the progress bar is red or else proceed to the next one. This way, the user can track their progress and can decide their next step based on quantitative data.

2.5.2. CT practices dashboard

The micro-interactions of the user are tracked using LogUI and notebook metadata and are mapped to the CT practices, as per Table 2. These are shown in 4 sections corresponding to the CT practices and each micro-interactions is displayed module-wise. A screenshot of the dashboard for one of the CT practices is shown in Figure 2.

2.6. Integration and Reproducability

The framework created for CT assessment in this research can be integrated and reproduced easily for any Python beginners course that uses Jupyter notebooks. The detailed instructions can be found on the Github repository (Agarwal, 2021). The steps to reproduce this CT assessment are:

1. Setup a LogUI server following the documentation (Maxwell and Hauff, 2021)
2. Add metadata tags to the course cells
3. Add the code for logging the micro-interactions into each notebook
4. Add the LogUI client files and configure the LogUI server link and authorisation token (follow LogUI client instructions)
5. Add the CT concepts dashboards to the modules and the overall CT practices dashboard (user ID to be configured here)

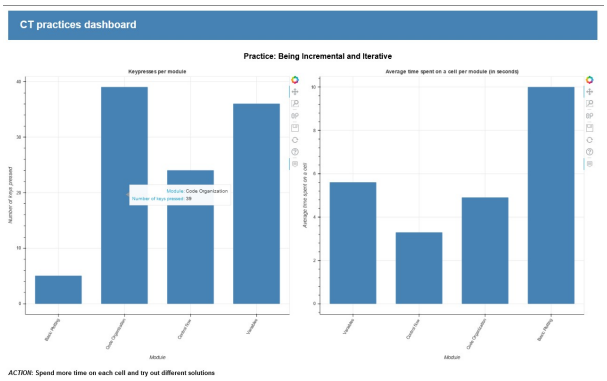


Figure 2. Global CT Practices dashboard

3. METHODS

To test the effectiveness of the Computation Thinking Assessment (CTA) framework developed, a user evaluation study was conducted for a period of length of 20 days.

25 participants signed up via an open call for participation, out of which 48% of the participants (12 participants) completed the study. Among the 13 participants who dropped out, 5 logged in but did not make much progress due to time constraints while 8 of them did not log in to the JupyterHub server at all. Only the 12 participants who completed the course are considered for further results and conclusions, thereby setting the sample size to 12. All the participants are in the age range 20-30 years. As part of the call for participation, the participants were asked to report their prior Python programming experience on a scale of 1-10, with 1 signifying 'no knowledge' and 10 signifying 'master'. 2 of the participants have moderate prior Python programming experience while 10 of them have no knowledge to little knowledge. Based on these characteristics, the user evaluation study considered participants who are beginners to Python programming at the university level from different domains.

The user begins by filling in the pre-evaluation survey and logging in to the JupyterHub server. They then fetch the modules from the server as assignments and complete the learning modules one at a time. The CT concepts dashboard is to be viewed after each module and provides feedback about whether the progress is satisfactory and if the module needs to be repeated. Once the user completes all the modules, they view the global CT practices dashboard for further overall feedback. Following that, the user fills in the post-evaluation

survey to assess their CT skills after learning basic programming.

An experimental design is used to measure the improvement in CT skills of participants before and after taking the Python basic programming course with the CTA framework integrated. Both these surveys have the same 24 questions with a five-point Likert scale (1=Strongly Disagree, 2=Disagree, 3=Neutral, 4=Agree, 5=Strongly Agree). The scores between the two surveys are compared to see the change in CT skills. The self-reported CT skills of users before and after taking the course are the dependent variable. A within-subjects design is chosen to measure the change in CT skills of each participants before and after taking the Python basic programming course. Based on this experimental design, the null hypothesis H0 and alternate hypothesis for the experiment H1 respectively are: *H0 = There is no difference in the CT skills users before and after taking the course, H1 = There is a difference in the CT skills of users before and after taking the course.*

The survey used is an adapted version of the survey created by Kılıç, Göko ̆glu, and Öztürk, (2021). This survey is used as it has been designed to evaluate the programming-oriented CT skills at the university level. As my research associates programming concepts with CT skills, it was necessary to find a survey scale that measures this correspondence same. This survey was found to be the best-suited to this purpose.

4. RESULTS

This research aims to answer the research question: *How can computational thinking be assessed through detection of user micro-interactions in a university-level self-paced Python beginners course integrated into Jupyter notebooks?*

To answer this research question, the results of the study are analysed under 3 research sub-questions :

1. RQ1: Did the users acquire Computational Thinking (CT) skills in the form of both CT concepts and CT practices?
2. RQ2: Was there a significant improvement in the self-reported CT skills of users after taking the Python basic programming course?
3. RQ3: How do the self-reported survey responses correspond to the actual user micro-interaction data?

4.1.1. RQ1: Did the users acquire CT skills in the form of both CT concepts and CT practices

To analyse the acquisition of self-reported CT skills, the post-evaluation survey was used. The count of each of the options of the Likert scale was aggregated per question for the 12 completed users, as shown in Figure 3. Following this, the mean (taken by encoding the Likert option values) and standard deviation (SD) was computed per question to get the final score per question, shown in Figure 3. Then, the average value of the mean for the CT concepts questions(15-24) and CT practices questions(1-14) was computed and was found to be 4.35 and 4.27 respectively. The standard deviation for the CT concepts questions(15-24) and CT practices questions(1-14) are both found to be in the range of 0.53-1.13, signifying a short deviation from the average value. Based on these values, it can be concluded that the users acquired CT skills in the form of both CT concepts and CT practices. As the self-reported survey questions pertain directly to the acquisition of Python programming skills, the value of the mean and SD also imply an improvement in Python programming skills of the user.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree	Mean	SD
I can detect errors related to coding on a subject I know.	0	0	1	6	5	4.33	0.82
I can detect and clean unnecessary code structures in a program.	0	0	4	5	3	3.92	0.58
I can detect the similarities and differences between two different programs.	0	0	3	5	4	4.08	0.59
I can understand how the result changes when different values assigned to the variables in the program.	0	0	0	4	8	4.67	1.13
I can continue the coding on a subject where I left off.	0	0	0	3	9	4.75	1.28
I can interpret the causes of the errors I encountered during the coding process.	0	0	2	4	6	4.33	0.75
I can detect errors in syntax (if, for, operators, etc.)	0	0	3	4	5	4.17	0.60
I can decide on the operations that will create my code structures or code blocks in the program	0	0	2	5	5	4.25	0.68
I can break a problem into smaller parts and work on them independently.	0	0	2	5	5	4.25	0.68
I can fix errors in the mathematical operations of the program.	0	1	0	6	5	4.25	0.85
I can understand an existing program and reuse it in my code.	0	0	1	5	6	4.42	0.84
I can detect and fix a logical error in the flow of a program.	0	0	4	4	4	4.00	0.53
I can sort the solution steps of the complex program appropriately.	0	0	2	6	4	4.17	0.71
I can visualize the processing steps of the program in my mind before I start coding.	0	0	2	5	5	4.25	0.68
I can use the loop structures (for, while, etc.) appropriately.	0	0	1	6	5	4.33	0.82
I can use the decision structure (if-else, switch-case) appropriately	0	0	2	4	6	4.33	0.75
I can determine the suitable data type (string, int, char, float, etc.) for a variable.	0	0	2	3	7	4.42	0.88
I can use mathematical (,-, ++ etc.) and logical operators (and, or, etc.) appropriately.	0	0	1	3	8	4.58	1.08
I can use general methods of programming languages [write(), read(), wait(), move(), scanf(), printf(), etc.]	0	0	3	5	4	4.08	0.59
I can code programs in which decisions (if-else, switch-case) and loops (for, while) can be used together.	0	0	0	6	6	4.50	0.98
I can make sections that require mathematical operations in the program.	0	0	0	7	5	4.42	0.99
I can determine which of the similar structures (if-switch, for-while) would be more appropriate.	0	0	1	5	6	4.42	0.84
I know which variables I would use before I start coding.	0	0	2	4	6	4.33	0.75
I can decide how to split complex programs into smaller pieces.	0	0	2	7	3	4.08	0.83

Figure 3. Mean and SD values per question

4.1.2. RQ2: Was there a significant improvement in the self-reported CT skills of users after taking the Python basic programming course?

To analyze the significance of the change in CT skills before and after the Python basic programming course, a statistical approach is used by conducting a paired t-test for the population. The paired t-test was done by considering the average of the responses in the pre-evaluation survey for each user and the average of the responses in the post-evaluation survey for each user as the pair of dependent variables. The null hypothesis H0 and alternate hypothesis H1 respectively are: *H0 = There is no difference in the self-reported CT skills users before and after taking the course, H1 = There is a difference in the self-reported CT skills of users before and after taking the course*.

The significance level α is set to a value of 0.05. If the two-tailed $p - value < 0.05$, the null hypothesis H0 is rejected. As seen in Figure 3, the p-value is less than α. Thereby, the null hypothesis H0 is rejected for the group - showing a significant improvement in self-reported CT skills. Based on the above results, it can be concluded that there is a significant change in the self-reported CT skills of users before and after taking the course.

4.1.3. RQ3: How do the self-reported survey responses correspond to the actual user micro-interaction data?

To answer RQ3, the average scores of the increase in self-reported CT skills were computed and compared to the user micro-interaction data and dashboard usage data.

As can be seen from Figure 5, the change in CT skills reported by the users roughly corresponds to the user micro-interaction data. For example, User 1 reports a high change of 3.4 and 3.2 in CT concepts and CT practices and this is reflected accordingly in the high values of the average CT concepts dashboard scores and runs and the

values of the CT practices dashboard. On the other end of the spectrum, low self-reported scores correspond to low values in the micro-interaction data. An example of such a user is User 2. From the data in Figure 5, it can be seen that User 4 reports a low change in the CT skills. This user has a good prior knowledge of the Python programming language(5 out of 10) and thereby did not gain much added value from the course. This user also scores highly on the CTC_DB_avg, signifying a good knowledge of the programming constructs and spend quite less time on the course, as is seen in the low 'Time spent' and 'Cell runs' in the CTP_DB_avg. Based on the correspondence between the self-reported survey responses and the actual user micro-interaction data, it can be concluded that they reflect quite strongly on each other, thereby implying honest responses to the survey questions.

```
Paired t-test

------------------  ------------
Sample size         12
Difference Mean      2.18403
t                    7.09089
Df                  11
P-value (one-tail)   1.00839e-05
P-value (two-tail)   2.01679e-05
Lower 95.0%          1.50611
Upper 95.0%          2.86194
------------------  ------------
```

Figure 4. Paired t-test result

User	CTC_S_avg	CTP_S_avg	CTC_DB_avg	CTC_DB_runs	CTP_DB_avg
User 1	3.4/4	3.2/4	8/10	2	Keystrokes: 438 Time spent: 113s Cell runs: 21.75 Errors: 0 Copy: 1 Paste: 1 Functions: 1.5 Modules imported: 3
User 2	1/4	0.93/4	7/10	1	Keystrokes: 19 Time spent: 6.5s Cell runs: 9.75 Errors: 0.75 Copy: 0.5 Paste: 1 Functions: 1.25 Modules imported: 3
User 3	1.9/4	1.7/4	7.68/10	1.25	Keystrokes: 96 Time spent: 21s Cell runs: 8 Errors: 0.25 Copy:0.5 Paste: 1 Functions: 1.25 Modules imported: 3
User 4	0.7/4	1.07/4	8.32/10	1	Keystrokes: 176 Time spent: 44s Cell runs: 5.75 Errors: 0.75 Copy: 0.5 Paste: 2 Functions: 1.5 Modules imported: 3

Figure 5. User micro-interaction scores and survey response changes

5. CONCLUSION

This research aimed to answer the research question - *How can computational thinking be assessed through detection of user micro-interactions in a university-level self-paced Python beginners course integrated into Jupyter notebooks?* To answer this research question, a framework for computational thinking (CT) assessment using detection of micro-interactions was developed and integrated in a university-level self-paced Python beginners course in Jupyter notebooks. A user evaluation study is conducted to show that this framework can be used to improve the acquisition of CT skills via an improvement in Python programming skills. To assess CT, a combination of a survey and portfolio assessment method are used in this research. The portfolio assessment is done by detecting user micro-interactions and using them as indicators of CT - providing a holistic view of the users' CT skills. As the portfolio assessment cannot capture the users' attitudes towards learning and affective outcomes, a survey is used before and after the programming course to assess these. The results show an improvement in CT skills of the users and an accurate assessment of the same through this framework. The results of the user evaluation study show that the developed framework for computational thinking (CT) assessment using detection of micro-interactions can be easily integrated in a university-level self-paced Python beginners course in Jupyter notebooks and this framework is effective in improving CT skills among users. In addition, a mapping of CT skills to the micro-interactions is developed in this research and this is used to create CT dashboards that provide feedback for self-regulation to users.

There are 2 main limitations of this research. Firstly, the results of the micro-interactions logging and the dashboard are not available to the user in the form of the global CT practices dashboard at all points of time. As the logging library - LogUI - is still in the development phase, it does not currently have the functionality to stream or access the user interaction logs in real time. This could cause issues in scaling as the number of users increases. The LogUI development team is currently working to resolve this issue and implement this functionality. The second limitation is that the assessment of self-regulated learning - Motivated Strategies for Learning Questionnaire (MSLQ) (Pintrich et al., 1991) - could not be fully integrated in this research owing to the time constraints of the user evaluation study. MSLQ is a self-reported questionnaire used to assess the cognitive view of motivations and learning strategies in a college course. Adding the MSLQ validation would help assess the self-regulated learning among students through this course. Owing to this limitation, the self-regulation aspect of this CT framework could not be fully assessed in this research.

In conclusion, a framework to assess CT skills was developed for a university-level self-paced Python beginners course and micro-interaction data was used to provide feedback to improve the acquisition of CT skills by the user. This framework can be integrated easily into other courses that teach CT skills through Python programming using Jupyter notebooks. While the user evaluation study conducted validates the CT assessment framework developed for a basic programming course, the results might differ for an advanced programming courses and courses that do not teach programming. Future work aimed at testing the applicability of this framework to other non-programming courses and to advanced programming courses should be carried out to validate the results of this CT assessment framework to them. In addition, integration of the MSLQ validation framework would enable validation of the complete theoretical design of this CT assessment framework.

6. REFERENCES

National Research Council & Others. (2010). Report of a workshop on the scope and nature of computational thinking. National Academies Press.

National Research Council & Others. (2011). Report of a workshop on the pedagogical aspects of computational thinking. National Academies Press.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. Computers & Education, 148, 103798. doi:10.1016/j.compedu.2019.103798

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada, 1, 25*.

Barr, V., & Stephenson, C. (2011). Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community*? ACM Inroads, 2*(1), 48–54. doi:10.1145/1929887.1929905

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Pei, C. (yu), Weintrop, D., & Wilensky, U. (2018). Cultivating Computational Thinking Practices and Mathematical Habits of Mind in Lattice Land. *Mathematical Thinking and Learning, 20*(1), 75–89. doi:10.1080/10986065.2018.1403543

Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2018). Preparing teachers to engage rural students in computational thinking through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education, 69*(4), 386–407.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175–192.

Hadad, R., & Lawless, K. A. (2015). Assessing computational thinking. In *Encyclopedia of Information Science and Technology, Third Edition* (bll 1568–1578). IGI Global.

Lyon, J. A., & J. Magana, A. (2020). Computational thinking in higher education: A review of the literature. *Computer Applications in Engineering Education, 28*(5), 1174–1189.

Kılıç, S., Gökoğlu, S., & Öztürk, M. (2021). A Valid and Reliable Scale for Developing Programming-Oriented Computational Thinking. *Journal of Educational Computing Research, 59*(2), 257–286.

Jivet, I., Scheffel, M., Schmitz, M., Robbers, S., Specht, M., & Drachsler, H. (2020). From students with love: An empirical study on learner goals, self-regulated learning and sense-making of learning analytics in higher education. *The Internet and Higher Education, 47,* 100758. doi:10.1016/j.iheduc.2020.100758

Schwendimann, B. A., Rodriguez-Triana, M. J., Vozniuk, A., Prieto, L. P., Boroujeni, M. S., Holzer, A., … Dillenbourg, P. (2016). Perceiving learning at a glance: A systematic literature review of learning dashboard research. *IEEE Transactions on Learning Technologies, 10*(1), 30–41.

Jivet, I. (2016). *The Learning tracker: a learner dashboard that encourages self-regulation in MOOC learners*. Opgehaal van http://resolver.tudelft.nl/uuid:f6c2ede4-a4e3-4ff0-b681-b0d057854e3c

Jivet, I., Scheffel, M., Drachsler, H., & Specht, M. (2017). Awareness Is Not Enough: Pitfalls of Learning Analytics Dashboards in the Educational Practice. In É. Lavoué, H. Drachsler, K. Verbert, J. Broisin, & M. Pérez-Sanagustín (Reds), *Data Driven Approaches in Digital Education* (bll 82–96). Cham: Springer International Publishing.

Broadbent, J., & Poon, W. L. (2015). Self-regulated learning strategies & academic achievement in online higher education learning environments: A systematic review. *The Internet and Higher Education, 27*, 1–13.

Yeni, S., & Hermans, F. (2019). Design of CoTAS: Automated Computational Thinking Assessment System. *perspectives, 23*, 28.

Hambrusch, S., Hoffmann, C., Korb, J. T., Haugan, M., & Hosking, A. L. (2009). A Multidisciplinary Approach towards Computational Thinking for Science Majors. *SIGCSE Bull., 41*(1), 183–187. doi:10.1145/1539024.1508931

in Education (ISTE), I. S. F. T., & (csta), C. S. T. A. (2011). *Operational Definition of Computational Thinking*. Opgehaal van https://cdn.iste.org/www-root/ct-documents/computational-thinking-operational-definition-flyer.pdf

Yuen, T. T., & Robbins, K. A. (2014). A Qualitative Study of Students' Computational Thinking Skills in a Data-Driven Computing Class. *ACM Trans. Comput. Educ., 14*(4). doi:10.1145/2676660

Matcha, W., Gašević, D., Jovanović, J., Uzir, N. A., Oliver, C. W., Murray, A., & Gasevic, D. (2020). Analytics of learning strategies: the association with the personality traits. *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, 151–160.

Maxwell, D., & Hauff, C. (2021). LogUI: Contemporary Logging Infrastructure for Web-Based Experiments. *Advances in Information Retrieval (Proc. ECIR)*, 525–530.

Pintrich, P. R., & Others. (1991). *A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ)*.

Agarwal, B. (n.d.). ct_dashboards. *GitHub*. Opgehaal van https://github.com/bhoom10/ct_dashboards

DOI: 10.34641/ctestem.2022.470

Solving Domain-Specific Problems with Computational Thinking

Sharon CALOR[1, 2*], Izaak DEKKER[1, 3], Dorrith PENNINK[1], Bert BREDEWEG[1, 4]
[1]Amsterdam University of Applied Sciences, the Netherlands
[2]Vrije Universiteit Amsterdam
[3]Erasmus University Rotterdam, The Netherlands
[4]University of Amsterdam, The Netherlands

*s.m.calor@hva.nl, i.dekker@hva.nl, d.h.m.pennink@hva.nl, b.bredeweg@hva.nl

ABSTRACT

Computational thinking (CT) skills are crucial for every modern profession in which large amounts of data are processed. In K-12 curricula, CT skills are often taught in separate programming courses. However, without specific instructions, CT skills are not automatically transferred to other domains in the curriculum when they are developed while learning to program in a separate programming course. In modern professions, CT is often applied in the context of a specific domain. Therefore, learning CT skills in other domains, as opposed to computer science, could be of great value. CT and domain-specific subjects can be combined in different ways. In the CT literature, a distinction can be made among CT applications that substitute, augment, modify or redefine the original subject. On the substitute level, CT replaces exercises but CT is not necessary for reaching the learning outcomes. On the redefining level, CT changes the questions that can be posed within the subject, and learning objectives and assessment are integrated. In this short paper, we present examples of how CT and history, mathematics, biology and language subjects can be combined at all four levels. These examples and the framework on which they are based provide a guideline for design-based research on CT and subject integration.

KEYWORDS

Computational thinking, Domain-specific problems, Developed examples, Integration, K-12

1. INTRODUCTION

Computational thinking (CT) was initially introduced by Papert (1980) as a method to perceive relationships between parts of a complex system. Wing (2006) defined CT as a way to solve problems, design systems and explain behavior by exploiting concepts from computer science. Shute et al. (2017) argue that such thinking can in principle also be done without computers, Denning and Tedre (2021) state that CT is in practice intertwined with its application in computers. Concepts from CT have also influenced the way in which we (from the viewpoint of different sciences) explain reality using information processing. Denning and Tedre (2021) therefore propose a twofold definition. On the one hand, CT consists of the ability to design applications that enable computers to perform tasks for us and, on the other hand, of the skills with which we can explain and interpret the world in terms of information processes. Defined in this way, CT is a set of skills essential to every modern profession in which the use of large amounts of data (information) is important. Typical activities associated with this concept in the literature include simulation, data mining, networking, automated data collection, gaming, algorithmic reasoning, robotics, programming, problem solving, modeling, data analysis and interpretation, as well as statistics and probability (Shute et al., 2017). These types of activity require several key skills that are linked in an iterative process: decomposition, abstracting, algorithmic thinking, debugging, iteration, and generalization (Shute et al., 2017).

2. CT-SUBJECT INTEGRATION

In K-12 curricula, CT skills are often taught in separate programming courses. However, literature on transfer of learning (Salomon & Perkins, 1989) suggests that without specific instructions, CT skills will not automatically transfer to other domains in the curriculum when they are developed while learning to program in a separate programming course. Integrating CT in existing courses could be of great added value because CT is often applied in practice in the context of a particular domain.

Yeni et al. (in press) distinguish three phases that are ideally completed when applying CT in a domain based on the process model of Barendsen and Bruggink (2019). In the first phase, a problem is converted into data or processes so that a computer can solve it. A computational solution is then created using an existing or self-developed program. Finally, the computational solution is re-interpreted in the context of the domain.

In addition, Yeni et al. (in press) classify the studies they located in their systematic literature review according to the degree to which CT skills are integrated with the subject-specific problems. At the substitution level, existing programs are applied by the teacher to illustrate a given matter. At the augmentation level, the students can use the programs themselves to find answers to questions without learning how the programs work as part of the lesson. At the modification level, the lesson design is different due to the use of CT: the learning objectives are no longer only focused on subject-related skills but enable students to adapt this subject with the help of CT. At the redefining level, students can use CT to solve questions/problems that cannot be solved without CT, for example, solving a problem by creating algorithms, simulations or programs. Therefore, at the highest level,

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

domain-specific problems are tackled that can only be solved with CT.

3. EXAMPLES

In this section, we present examples of how history, mathematics, biology and language subjects can be combined with CT at all four levels. These examples and the framework on which they are based provide a guideline for design-based research on CT and subject integration.

3.1. History

Examples of domain-specific problems in history that require CT are questions such as "What role did slavery play within the Dutch East India Company, and how did this role change during the 17th, 18th and 19th centuries?" or "How often were slaves recorded in notarial deeds during the 16th, 17th, and 18th centuries?"

Using Artificial Intelligence the National Archives of the Netherlands have digitally transcribed more than 2 million pages of 17th, 18th and 19th century texts from (among others) the Dutch East India Company and made them publicly available.

On the substitution level, generated data can be used for illustration purposes in the classroom. On the augmented level, students can search the database themselves on the basis of detailed searches and hypotheses to test hypotheses regarding colonial history. On the modification level, students can study if and how the database takes into account how language use changes over time. On the redefinition level, students can formulate and test hypotheses using the database and search strings that consider changes in language use over time.

3.2. Mathematics

An example of a domain-specific problem in mathematics that requires CT is "How to interpret and analyze large datasets?"

Quantifying and visualizing data obtained with the help of statistical software and making statements in the field of explanatory statistics based on such data has CT potential.

On the substitution level, statistical software can be used to illustrate what the median or mode is in a large dataset. On the augmented level, students can process data from large datasets into an appropriate table or graph and test it for value. On the modification level, students can make statements about a population based on sample data and quantify its reliability. On the redefinition level, students can design a plan to obtain answers to a problem statement using large datasets, connect interpretations to the obtained data and interpret the result in terms of the context.

3.3. Biology

An example of a domain-specific problem in biology that requires CT is "How can we, e.g., track and explain biodiversity loss with respect to the bee population?"

The Global Biodiversity Information Facility (http://gbif.org) makes large datasets available that include information on, for example, the diet and reproduction of bees in various European cities.

On the substitution level, the results of studies that employ large datasets are used as examples to support biodiversity theory. On the augmentation level, the teacher formulates questions on the basis of biodiversity theory that students can answer using a dataset. On the modification level, the teacher demonstrates which code can be used to analyze a dataset in Python to answer questions and allows his or her students to practice with this dataset. On the redefinition level, students may modify the code to answer new questions using the available dataset.

3.4. Language

An example of a domain-specific problem in language is "How do you find the most relevant and reliable information for an argument from a nearly infinite dataset of sources?"

Data retrieval is a technique with which data can be efficiently extracted from large datasets. To utilize this potential, it is necessary to formulate search strings with which the search engine can make targeted selections.

On the substitution level, students can use search engines to replace library catalogs (search by author, source, genre). On the augmented level, students can enter and test predefined search strings (search terms that are linked with Boolean operators AND, OR or NOT) in an online database. On the modification level, students can assess which parts of the search string are not functioning properly and require replacing. On the redefinition level, students can formulate, test and modify in iterations a search string that excludes and includes exactly the desired resources from a large dataset.

4. FUTURE RESEARCH

In this short paper, we have presented examples of how history, mathematics, biology and language subjects can be combined with CT on four levels. These examples, and the framework on which they are based, provide a guideline for design-based research on CT and subject integration.

Our ongoing work implements the described examples in the classrooms in which the different subjects are taught.

5. AUTHOR AND CONTRIBUTOR STATEMENT AND DATA ACCESS STATEMENT

5.1. Author and Contributor Statement

5.1.1. Author Statement

Sharon M. Calor: Conceptualization, Writing – Review & Edit

Izaak Dekker: Conceptualization, Writing – Review & Edit

Dorrith H.M. Pennink: Conceptualization, Writing – Review & Edit

Bert Bredeweg: Conceptualization, Writing – Review & Edit

5.2. Data Access Statement

Not applicable.

6. REFERENCES

Barendsen, E., & Bruggink, M. (2019). Het volle potentieel van de computer leren benutten: over informatica en computational thinking. *Van Twaalf tot Achttien*, *29*(10), 16-19. https://onderwijstijdschriftenplein.nl/tplein/van-twaalf-tot-achttien-jrg-29-december-2019-nr-10/

Denning, P. J., & Tedre, M. (2021). Computational thinking: A disciplinary perspective. *Informatics in Education*, *20*(3), 361-390. https://doi.org/10.15388/infedu.2021.21

Salomon, G., & Perkins, D.N. (1989). Rocky roads to transfer: Rethinking mechanisms of a neglected phenomenon. *Educational Psychologist*, *24*(2), 113–142.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142-158. https://doi.org/10.1016/j.edurev.2017.09.003

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. http://www.cs.cmu.edu/~wing/

Yeni, S., Grugrina, N., Hermans, F. F. J., Tolboom, J. & Barendsen, E. (in press). Embedding computational thinking in the non-computing subjects: A systematic literature review.

DOI: 10.34641/ctestem.2022.452

Precoding skills - Teaching computational thinking to preschoolers in Singapore using unplugged activities

Vidhi Singhal (Founder, Kinder Koder), Singapore, vidhi@kinderkoder.com

ABSTRACT

Computational Thinking (CT) by now is widely recognized as an important skill in K-12 education. Research suggests that, during children's early formative years, certain types of experiences, including exploration, exposure to basic skills, and practice with rich communication, among others, are critical to support typical development (Ramey and Ramey 1999). Exposure to these experiences cultivates school readiness, which in turn supports children's later achievement. The same holds true for CT. Exposing children to different kinds of CT-oriented problem-solving ideas and strategies, paired with thoughtful guidance, will allow preschool-aged children to practice CT over a wide variety of contexts. Kinder Koder was started in 2020 with the aim of teaching CT to preschoolers through unplugged games and activities. This paper shares how CT is being taught in Kinder Koder's enrichment classes in Singapore by using a framework more suited for early childhood education. This will help preschool teacher's integrate teaching CT in their day to day classrooms.

KEYWORDS

computational thinking, unplugged, kindergarten, play based learning

1. INTRODUCTION

Coding is, "telling the computer what to do and how to do it." Before you can think about coding, you need to work out exactly what you want to tell the computer to do. Thinking through problems this way is Computational Thinking. It allows us to take complex problems, understand what the problem is, and develop solutions. So CT is the step that comes before you actually do coding and hence we call it precoding skills to make it simpler for parents to understand. Figure 1 shows this relationship between CT and coding.

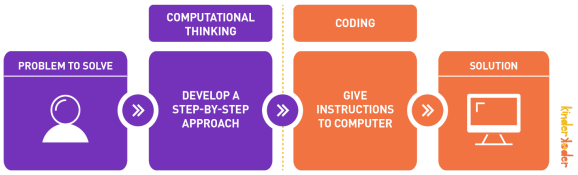


Figure 1. CT vs Coding

2. FRAMEWORK

Computational thinking involves a broad set of approaches and skills. As per ISTE's operational definition Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:

•Formulating problems in a way that enables us to use a computer and other tools to help solve them.

•Logically organizing and analyzing data

•Representing data through abstractions such as models and simulations

•Automating solutions through algorithmic thinking (a series of ordered steps)

•Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources

•Generalizing and transferring this problem solving process to a wide variety of problems

Abstraction as a concept is difficult for preschoolers. Computational thinking is itself a very abstract idea for kids and it can be made less abstract by teaching them unplugged, out of the screen, into the physical world. Giving children something to hold with their hands, like blocks or cards. This stimulates active engagement and allows children to experience the material. Instead of speculating about what would go wrong, they can try it themselves and experience the consequences of certain actions. To make learning for age appropriate we've come up with the following framework (Figure 2) for early childhood i.e. 3-6 years.

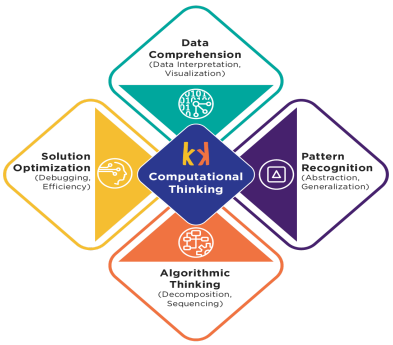


Figure 2. CT Framework

3. OUR APPROACH

For each of the skills we have identified age appropriate learning outcomes and we develop unplugged activities to support that learning as shown in Table 1.

Table 1. Example of age appropriate learning outcomes

CT Skill	Age (3-6 years)
Data comprehension	• Understanding directions (left, right, forward) • Sorting of numbers (1-10) • 1:1 correspondance

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Pattern recognition	• Grouping of similar objects basis shape, size, color • Being able to identify, extend and create simple patterns
Algorithmic thinking	• Follow instructions to complete simple tasks (drawing, coloring) • Understanding what algorithms are • Focus on sequence of steps
Solution Optimization	• Understanding a problem can have multiple solutions and some better than the other • Understanding what is an error

4. IMPLEMENTATION

4.1 Data comprehension

Logically organizing, representing and interpreting data. Understand instructions and numbers. E.g. Figure 3:

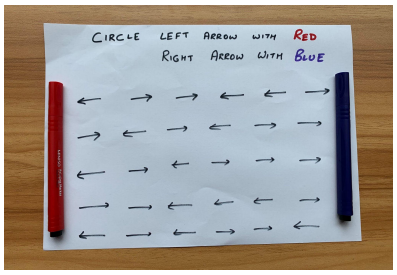


Figure 3. Sample activity for data comprehension

4.2 Pattern Recognition

Pattern recognition is the process of identifying, defining, extending, and creating patterns. This forms the foundation for higher order thinking skills such as abstraction (hiding the complexities of one pattern from another) and generalization (spotting things that are common between patterns). E.g. Figure 4:

Figure 4. Sample activity for pattern recognition

4.3 Algorithmic Thinking

To break down a problem into smaller sub-problems. This is known as Decomposition. Then finding a step by step solution to a sub problem.E.g. Figure 5:

Figure 5. Sample activity for algorithmic thinking

4.4 Solution Optimization

To be able to identify gaps in solutions, evaluate, resolve inconsistencies, optimize and come up with an efficient solution. E.g. Figure 5:

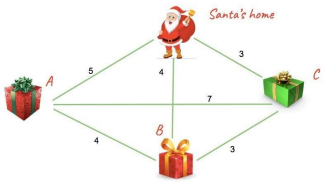


Figure 5. Sample activity for solution optimisation

5. CONCLUSION

Over 300 students so far have benefited from this approach. While we lack quantitative data to measure their learning outcomes, the qualitative responses have been very promising. Both parents and kids have found this way of teaching very engaging and have repeatedly asked for more content. Feedback from a parent below:

"Amazing, Amazing, Amazing. Ma'am, your ideas are so unique. They are just lovely. Thank you for sharing them." Parent of a preschooler.

This unplugged approach introduces CT concepts to kids in a non daunting way making them more prepared for future learning. As a next step we would look for ways to measure the learning outcomes.

6. REFERENCES

CAS-UK.Computing at School Working Group http://www.computingatschool.org.uk

Computer Science Unplugged https://www.csunplugged.org/en/

ISTE and CSTA(2007). Operational definition of computational thinking for K-12 education.

Ramey, Craig T., and Sharon L. Ramey (2004). "Early Learning and School Readiness: Can Early Intervention Make a Difference?" Merrill-Palmer Quarterly 50, no. 4: 471–91

Wing,J.M.(2006).Computational thinking Communications of the ACM, 49(3), 33-35

DOI: 10.34641/ctestem.2022.453

Computational Thinking in Language Arts When Teaching Creative and Expository Writing

Aysegul Bayraktar, Yasemin Gulbahar
DAnkara University, Turkey
abayraktar@ankara.edu.tr; ysmnglbhr@gmail.com

ABSTRACT

In language arts instruction, computational thinking skills can be implemented through language arts activities, especially in teaching of diamante poems and expository writing. Instructional materials and activities for an elective language arts course lasting 14 weeks for two hours per week were prepared by the researcher. During the course, an invited speaker presented one instructional session for 23 pre-service elementary school teachers regarding the 'Scratch Program' used to create digital stories. The following week another expert instructed computational thinking skills to these pre-service teachers about how to implement these skills into their language arts activities. Additionally, the researcher provided three hours and 30 minutes of instruction regarding poetry writing and 10 hours instruction on expository writing to increase the pre-service teachers' abstraction, separation, pattern recognition, logical reasoning, pattern decomposition, error detection, and algorithm design skills. Furthermore, the pre-service teachers were trained regarding various writing genres including creative writing. The pre-service teachers were later asked to submit a portfolio of their writing samples and the activities prepared specifically for developing elementary school students' computational thinking skills along with their reflective journals written regarding their experience of learning computational thinking skills. The pre-service teachers were not having or realizing any computational thinking skills in the beginning of the semester. Whereas when the semester ended, their reflective journals and written samples from portfolios showed they had become knowledgeable about computational thinking skills as well as strategies and/or activities used to increase their future students' computational thinking skills.

KEYWORDS

Preservice teachers, computational thinking, poetry writing, expository writing

1. INTRODUCTION

Computational thinking (CT) can be explained as having a mental automated solutions (Yadav, Mayfield, Zhou, Hambrusch, & Korb, 2014). CT includes skills like abstraction, separation, pattern recognition, logical reasoning, pattern decomposition, error detection, and algorithm design skills. Recently, it is argued that CT can be taught at early ages starting from preschool education. However, in order for teachers to be knowledgeable enough to teach CT skills to their students they need to gain appropriate training on developing CT skills during their undergraduate education. This paper explains 23 elementary school preservice teachers' experiences whom were registered for an elective language arts course lasting 14 weeks for two hours per week. The participating pre-service teachers were taught several writing genres including journal, personal, story, poetry, expository, persuasive writing as well as some software programs like 'Scratch' and features and principles of computational thinking skills.

2. PROCEDURE

While teaching poetry and expository writing, the goals were improving pre-service teachers' computational thinking skills especially on abstraction, separation, pattern recognition, logical reasoning, pattern decomposition, error detection and algorithm design. In order to reach these goals some educational activities were designed and implemented in line with the goals determined by the Ministry of National Education (MoNE). Some of these goals were determining the story elements in the texts (The subject of the text, plot, location, time, characters); explaining the contribution of nouns and adjectives to the meaning of the text; realizing the meanings of verbs; distinguishing text types; using information sources effectively; providing information on how to use the contents and glossary in printed and digital content to access information; and writing expository text.

2.1. Poetry Writing – Writing a Diamante Poem

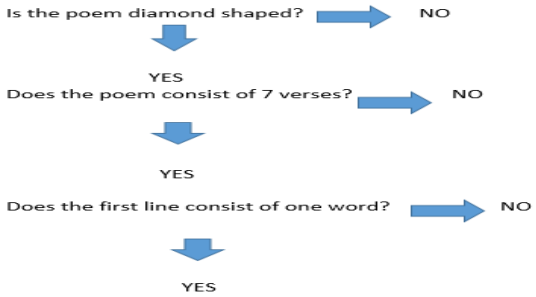
The activities pre-service teachers did on creative writing and poetry writing started with listening to a story. For developing abstraction skills, pre-service teachers listened to and talked about a story. The instructor chose and read a children's picture book in which participants heard several examples of nouns, adjectives, and verbs. The title of the story was not shared with learners. After the story was read, the instructor asked WH questions for them to comprehend the story completely.

To work on abstraction, logical reasoning, and pattern recognition skills, the participants found appropriate titles, adjectives, verbs, and nouns for the story. After the participants correctly answered all the detail questions regarding the story, the instructor asked them to find an appropriate title for the story. Responses of them were taken and written on the board without filtering any responses. Later, as a class, they determined an appropriate title. The instructor asked students and prospective teachers which adjectives they heard in the story. Heard adjectives were listed on the board too. Then, pre-service teachers were asked to think about other adjectives that can be used in the story instead of the listed ones. The same procedure was repeated for other types of words (nouns and verbs). Therefore, they brainstormed again to find different nouns, adjectives and verbs that could be used and were related to the content of the story.

For logical reasoning skills, pre-service teachers chose objects can be seen and used in the classroom. In this activity, they were expected to drag the objects they see in the classroom to the school picture.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

By applying CT in language arts the authors aimed that pre-service teachers could identify and represent patterns in different sentences (Mishra et al., 2013). The instructor gave a presentation about different kinds of poems including acrostic, found, concrete, diamante, and etc. and asked to take notes of the prominent features of the poetry genres especially the found poems in order to improve their pattern recognition skills. Based on the rules and examples they saw pre-service teachers selected non-examples which was increasing their error detection skills. Then, they were asked to write an algorithm design for the diamante poem, similar to one given below:



At the end, they were asked to write their own diamante poem, in which they worked on abstraction and pattern recognition skills.

2.2. *Expository Writing - Learning about Fossils*

To increase pre-service teachers' pattern recognition skills the instructor showed two images (one realistic and one imaginal) and asked pre-service teachers about what they thought; were they similar or not? Which one could be fictional or non-fictional? And Why? Then, she shared some statements and asked which ones were the features of expository writing and listed some text types like comics, directions, text books, recipes etc. and asked which ones can be considered as non-fiction text. She also explained the differences between facts and opinions. The participants were given several statements and asked to determine which ones were facts. By doing these three activities, the instructor aimed to increase their logical reasoning skills.

Later, they watched a video about fossils. After seeing the video, the instructor emphasized the importance of the order of the steps in the formation of fossils and pointed out that if this order was not followed correctly, fossils would not be formed. Then, instructor asked them to create and write an algorithm design, in which they described the process and steps regarding the formation of fossils. Thus, they could work on algorithm design skills and since they were summarizing the process of forming fossils they were also increased their abstraction skills as one of the computational thinking skills. Participants did a research on fossils and by doing this they worked on pattern decomposition and decomposition skills. Then, they composed an expository text regarding fossils by using some of the transitions words given them in the classroom, which can be considered as increasing their abstraction skills.

3. CONCLUSION

The pre-service teachers were asked to submit a portfolio of their writing samples and the activities prepared specifically for developing their future elementary school students' computational thinking skills along with their reflective journals written regarding their experience of learning computational thinking skills. When the semester began, the pre-service teachers stated not knowing about computational thinking skills. Whereas when the semester ended, their reflective journals and written samples from portfolios showed they had become knowledgeable about computational thinking skills as well as strategies and/or activities used to increase their future students' computational thinking skills. In her reflected journal one pre-service teacher wrote: "*I find it very useful to get information about how I can give a critical and an inquiring perspective to my students.*" Another pre-service teacher stated "*I think it is really useful to learn where computational thinking comes into play, especially in the activities of folding paper, educational board games, educational games, etc. We have learned where students use computational thinking skills and where we can apply them better.*" Another pre-service teacher mentioned the importance of CT in solving daily life problems by stating "*Considering that the problems also exist in daily life, we should not ignore that the computational thinking will make our life easier.*" and making learning permanent "*the computational thinking concept will not only make students active during the lesson, but also increase the permanence of the learning.*" In line with the recommendations in NRC report (NRC, 2010) pre-service teachers written activities placed in their portfolios and reflected journals showed that introducing CT skills in fiction and nonfiction writing effectively influenced pre-service teachers' understanding of CT concepts.

4. REFERENCES

Mishra, P., Yadav, A., & Deep-Play Research Group. (2013). Rethinking technology & creativity in the 21st century. *TechTrends, 57*(3), 10-14.

National Research Council (NRC). 2010. Report of a workshop on the scope and nature of computational thinking. The National Academies Press.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE), 14*(1), 1-16.

DOI: 10.34641/ctestem.2022.454

Exploring Embedded Computational Thinking in STEM Teacher Education

Dorrith PENNINK[1*], Izaak DEKKER[1, 3], Sharon CALOR[1, 2], Monique PIJLS[1], Bert BREDEWEG[1, 4]
[1]Amsterdam University of Applied Sciences, the Netherlands
[2]Vrije Universiteit Amsterdam
[3]Erasmus University Rotterdam, The Netherlands
[4]University of Amsterdam, The Netherlands

*d.h.m.pennink@hva.nl, i.dekker@hva.nl, s.m.calor@hva.nl, m.h.j.pijls@hva.nl, b.bredeweg@hva.nl

ABSTRACT

Computational thinking (CT) has become a necessity in many professional domains. As such, scholars argue that the acquisition of CT and application should be embedded in existing school subjects. Within the CT literature, a taxonomy distinguishes CT practices in STEM education into four categories: data related, systems thinking, modeling & simulation and computational problem solving (CPSP). Practical applications of these different categories are still limited. This paper presents three examples in which educators of science teachers integrate CT within STEM content knowledge using the above mentioned taxonomy. The first example applies to CPSP and data practices, the second to CPSP exclusively, the final to systems thinking and modeling & simulation. The examples provide practical insight that makes the use of CT in STEM education more tangible for practitioners.

KEYWORDS

Computational Thinking, Teacher Education, STEM Education, Computational Practice, TPACK

1. INTRODUCTION

Digital elements are embedded in every aspect of our society. The Dutch educational system has not yet responded to this development. In the Netherlands, computational thinking (CT) is currently not a formal part of K-12 curricula. While plans are being drawn to include CT in curricula, teacher education needs to equip pre- and in-service teachers with the knowledge and skills to execute these planned innovations.

When classroom activities are becoming computational endeavors, a theoretically grounded operationalization of CT is required that describes the form it should take in STEM classrooms. Weintrop et al. (2016) meet this need, proposing a taxonomy in which four categories of practices are distinguished and illustrated: data related-, systems thinking-, modeling & simulation-, and computational problem solving practices (CPSP).

2. EMBEDDED CT

This paper presents three examples in which educators of pre- and in-service science teachers at the Amsterdam University of Applied Sciences (AUAS) integrated CT within STEM content knowledge using Weintrop et al.'s (2016) taxonomy.

2.1. Retrieving information from large datafiles

> *The know-it-all has an answer to every question, the wise asks the right questions.* (Bais, 2007)

In a robotics/CT-course, part of the main phase of the bachelor program for pre- and in-service STEM teachers, students were trained to use JavaScript and Python. As a skills training and versatility test, they were asked to retrack information from a COVID-19 dataset (World Health Organization, 2021) and create graphical representations of cases and deaths for different countries in a python notebook environment. Despite limited experience, students proofed capable of performing the task (Figure 1). Particularly interesting was the eagerness students demonstrated to find answers to a number of self-formulated questions. An animated societal and ethical discussion followed.

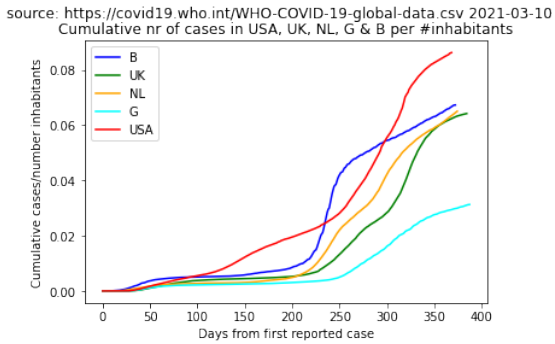


Figure 1. Graphical representation task based on WHO-COVID dataset

Practicing dataset skills with a specific dataset showed an un foreseen educational gain. Students surpassed the task-given boundaries and became architects of a personally relevant learning process. We regard this as an added value of integrating CT and content knowledge: creating a learning environment in which meaningfulness is shaped by learners themselves using CT-tools to answer their own questions.

2.2. Use of a CT-design map

In the same robotics/CT course students were trained to design programs for several tasks, partly self-defined.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Although usually capable of performing such tasks, students failed to explicitly identify key concepts of CPSP in their code. To address this problem the CT-design map was introduced in order to disentangle key-concepts. Figure 2 is (part of) a student made example. The top row shows the task description, following rows contain a description of subproblems with associated code lines and an explanation and/or description of action(s) performed. Bottom rows (not in figure) describe testing and debugging processes.

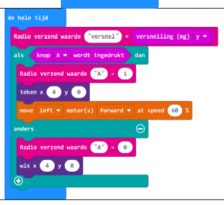
Taak-omschrijving	Ontwerp een programma dat dient als afstandsbediening voor de Bit:Bot. De afstandsbediening kan ten alle tijden gebruikt worden, zodat de besturing van de Bit:Bot overgenomen kan worden. • Als 'A' wordt ingedrukt, dan gaat de Bit:Bot naar rechts • Als 'B' wordt ingedrukt, dan gaat de Bit:Bot naar links • Als 'A + B' wordt ingedrukt, dan gaat de Bit:Bot rechtdoor	
Onderdeel	**Programmaregels**	**Toelichting**
Radiofrequentie en drempelwaarde juist instellen.		Als de Micro:Bit wordt aangezet, is het nodig om te schakelen naar groep 1 (zoals ook bij alle andere opdrachten het geval was).
Eerste mogelijkheid; Knop 'A' wordt ingedrukt		Als 'A' wordt ingedrukt, dan wordt er een signaal naar de Bit:Bot verstuurd waardoor de linker motoren gaan draaien op een snelheid van 60%. Als gevolg hiervan zal de Bit:Bot rechts afslaan. Als 'A' wordt losgelaten, dan zal de Bit:Bot autonoom verder rijden en treedt de lichtsensor weer in werking.

Figure 2. CT-design map – student's work

Using this map has several advantages. Firstly, it stimulates CPSP in a structured manner. Secondly, key facets of CT, such as abstraction, decomposition, pattern recognition and algorithmic thinking are clearly distinguishable. This contributes to the development of consciously competent teachers. An additional merit is offered by the 'explanation' column. In science education we emphasize the value of 'thinking-back-and-forth' between representations. Students are trained in 'talking-science', a type of storytelling to describe phenomena from different perspectives. The articulation of code is an expression of this form of thinking.

2.3. Systems thinking approach of modeling

All models are wrong, some are useful. (Anonymous)

With the ubiquity of covid-, climate- and other models, the societal relevance of dynamic modeling is indisputable. By embedding modeling in science education students learn about the possibilities and limitations of models. Furthermore, dynamical models are used to enhance conceptual understanding and test hypotheses. A complication in teaching modeling is the high cognitive load undermining learning effects (Van Buuren, Heck, and Ellermeijer, 2016). To cope with this problem a new strategy is explored in a kinematics and dynamics course in physics teacher education at AUAS. Before confronting students with modeling tasks, models are presented as concept maps constructed with modeling software.

The map in Figure 3 represents the relation between two of Newton's laws and kinematic quantities. As in the previous example students are encouraged in thinking-back-and-forth between the different perspectives (i.e. concept map, graphs, equations, movement description etc.).

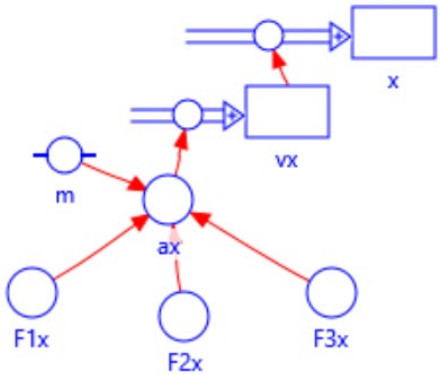


Figure 3. Systems thinking concept map

With this approach we plan to contribute to a better understanding of related concepts in mechanics and increase the learning efficacy of modeling tasks.

3. FUTURE WORK

In the near future embedded acquisition and application of CT will be part of existing school subjects. At the AUAS, science teacher education is preparing for this change with the introduction of a CT-curriculum. We explicitly articulate CT knowledge and skills, we investigate further CT enrichments embedded in STEM content knowledge, and we study, develop and apply the pedagogy of a subject-integrated approach. Our intention is not to deliver IT experts, but innovative professionals instead, with sufficient experience and self-confidence to implement the curriculum revision in a meaningful way in their own teaching practice.

The presented examples demonstrate our explorations to develop a CT-content knowledge integrated curriculum.

4. REFERENCES

Bais, S. (2007). *De sublieme eenvoud van relativiteit: een visuele inleiding*. Amsterdam University Press.

Van Buuren, O., Heck, A. & Ellermeijer, T. Understanding of Relation Structures of Graphical Models by Lower Secondary Students. *Res Sci Educ* **46,** 633–666 (2016). https://doi.org/10.1007/s11165-015-9474-x

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of science education and technology*, *25*(1), 127-147.

World Health Organization. (2021) Coronavirus COVID-19 daily new and cumulative cases and deaths by country. https://covid19.who.int/WHO-COVID-19-global-data.csv

DOI: 10.34641/ctestem.2022.471

An Exploratory Study of the Relationship between Computational Thinking and Creative Attitudes among University Students

Masanori FUKUI*[1], Yuji SASAKI[2], Tsukasa HIRASHIMA[3]
[1]Center for University Education, Tokushima University, Japan
[2]Graduate School of Media and Governance, Keio University, Japan
[2]Bridge UI Inc., Japan
[3]Graduate School of Technology, Hiroshima University, Japan
f-masanori@tokushima-u.ac.jp, y.sasaki@keio.jp, tsukasa@lel.hiroshima-u.ac.jp

ABSTRACT
This study aimed to explore the relationship between the computational thinking scale (CTS) and the creative attitudes scale among university students. A total of 93 university students were tested on the CTS ("creativity," "algorithmic thinking," "cooperativity," "critical thinking," and "problem solving") and the scale of creative attitudes ("flexibility," "analytical problem solving," "entrepreneurship," "perseverance," "imagination," and "co-operation"). The results show a significant correlation between the majority of CTS factors and creative attitudes. However, imagination and co-operation are only correlated with one or two CTS factors. Therefore, we identified factors of CTS and the creative attitudes that are related to each other.

KEYWORDS
computational thinking, creativity, creative attitudes, university students, empirical study

1. INTRODUCTION

1.1. Purpose of This Research

This study aimed to explore the relationship between computational thinking and the creative attitudes of university students and obtain basic knowledge for future class design.

1.2. Background of This Research

The information environment is advancing further, with the increase of advanced information technology and artificial intelligence (AI) and the Singularity's predicted arrival (Kurzweil, 2005).

In education, many projects have been conducted, including implementing programming education and training AI engineers. Therefore, the importance of fostering computational thinking has been emphasized (Wing, 2006).

The term computational thinking was first used by Papert (1980), however, Wing's essay led to the recognition of computational thinking as a basic problem-solving skill useful to all (Wing, 2014). Computational thinking is a set of problem-solving methods that involve expressing problems and their solutions that can be executed by a computer. Other definitions of computational thinking include the International Society for Technology in Education (ISTE) and the Computer Science Teachers Association's (CSTA) operational definition (2011) and more (e.g., Selby & Woolard, 2014). Because many of the current definitions commonly used computational thinking as defined by Wing (Shute et al, 2017), it has been used in this study.

For example, in the U.K., computing is a subject that focuses on developing computational thinking (Gov.UK: Department for Education, 2013). It consists of skills in problem-solving formulations, logical organization, and data analysis using computers and other tools, confidence in handling complexity and perseverance in tackling challenging problems, and thinking and expression of abstraction and algorithm design. These skills are not demonstrated when creating programs, however it can be applied to problem solving in all aspects of life. Moreover, many practices enhance computational thinking. For instance, computing at school led to the creation of teacher resources and implementation of many practices (Computing at School, 2015). There are many other practices to foster computational thinking (Fagerlund et al., 2020; Grover & Pea, 2015; International Society for Technology in Education (ISTE), n.d.). Moreover, Computational Thinking with Scratch by Harvard University focused on its development using Scratch and suggested evaluation methods (Harvard University, n.d.; Brennan & Resnick, 2012). In the U.K., progression pathways were proposed (Computing at School, 2015a).

In addition, it was highlighted that creativity is related to computational thinking (e.g., Doleck et al., 2017; Rotem et al., 2020). In this study, we focused on enhancing creativity and computational thinking. The possibility of enhancing creativity by increasing computational thinking or enhancing computational thinking by increasing creativity, makes computational thinking training promising. Therefore, the implementation of classes that realize computational thinking and creative development is necessary for future education.

1.3. Identification of Problems

The relationship between creativity and computational thinking was determined (Hershkovitz et al., 2019) in

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

several studies. In addition, there are various ways to assess creativity and computational thinking, such as evaluating portfolios and artifacts, measuring creativity directly with creativity tests, and examining the relationship between computational thinking and creativity using psychological scales.

However, for students who never demonstrated creativity or who had not experienced any education to enhance creativity, it is essential to understand whether they have an attitude to solve problems before attending computational thinking or special classes to demonstrate creativity. It is crucial to understand the relationship between computational thinking and creativity as readiness, and also to develop an attitude of creativity that is independent of a specific problem.

A creative attitude implies not following a set method or pattern of problem solving, however it requires posing questions, being curious and unafraid to constantly improve and create something new out of failure (Schank & Childers, 1988).

Subject matter can be developed, and appropriate lessons designed based on the understanding of the relationship between creative attitudes and computational thinking. Therefore, understanding this relationship is fundamental. However, no previous studies have addressed this.

In this study, we explored the relationship between computational thinking and the creative attitudes of university students as primary data for designing classes to enhance computational thinking and creativity.

2. RESEARCH METHOD

2.1. Survey Targets and Procedure

In November 2021, a survey was administered to second-year university students majoring in game development in Japan in a class taught by one of the authors. In the survey, responses were obtained from 93 participants (average age: 19.24, SD = 0.71, 88 males, 5 females). The effective response rate was 100.0%. The duration of the survey was approximately 15 min. As ethical consideration, in conducting the survey, there were no questions on personally identifiable information such as name, initials, e-mail address, or student ID number. The survey content was explained to the respondents prior to administration. Furthermore, they were advised that they should respond to the survey, only if they agreed with its content, and that their responses would be considered as their consent. The acquired data are encrypted and stored in a lockable location at the applicant's institution with restricted access.

2.2. Measurement Scales

To measure computational thinking, we prepared five factors and 29 items on the computational thinking scale. (Bando & Motozawa, 2021). This scale is a Japanese translation of the computational thinking scale developed by Korkmaz et al. (2017). Hereafter, the Japanese version of the computational thinking scale is denoted as "CTS." These five factors are: creativity, algorithmic thinking, cooperativity, critical thinking, and problem solving. The CTS is shown in Table 5 in the Appendix.

In addition, creative attitudes were measured using a revised version of the creative attitudes scale (Shigemasu et al., 1993) that contained six factors and 74 items. These six factors are: flexibility, analytical problem solving, entrepreneurship, perseverance, imagination, and cooperation. The measurement scale of creative attitudes is shown in Table 6 in the Appendix.

A five-point Likert scale (1–5) was used: "5: Strongly Agree, 4: Agree, 3. Undecided, 2: Disagree, 1: Strongly disagree."

Both of these scales were used in surveys of university students, and their usage is valid for this study.

2.3. Analysis of Procedure

First, descriptive statistics of the computational thinking scale and creative attitudes were calculated. Subsequent to confirming normality, the correlation coefficients between the factors and CTS items and creative attitudes were calculated.

3. RESULTS

3.1. Descriptive Statistics

Tables 1 and 2 show the results of the descriptive statistics of the computational thinking scale and creative attitudes. These results showed that the mean scores for all items and factors were above a medium score of 3.00.

Table 1. Descriptive Statistics of the Computational Thinking Scale

	Mean	*S.D.*
creativity	3.52	0.65
algorithmic thinking	3.19	0.81
cooperativity	3.72	0.90
critical thinking	3.23	0.73
problem solving	3.01	0.70

(n = 93)

Table 2. Descriptive Statistics of Creative Attitudes

	Mean	*S.D.*
flexibility	3.01	0.62
analytical problem solving	3.33	0.63
entrepreneurship	3.51	0.66
perseverance	3.51	0.69
imagination	3.44	0.60
cooperation	3.36	0.48

(n = 93)

3.2. Normality Test for the Computational Thinking Scale and Creative Attitudes

We tested the normality of each of the CTS and creative attitude factors using the Shapiro-Wilk test. The results are shown in Table 3.

Table 3 shows that in the CTS, normality was observed in creativity (W = 0.99, *n.s.*), algorithmic thinking (W = 0.98, *n.s.*), critical thinking (W = 0.98, *n.s.*), and problem solving

(W = 0.99, *n.s.*), but not in cooperativity (W = 0.95, p < .01). Normality for creative attitudes was found: flexibility (W = 0.99, *n.s.*), analytical problem solving (W = 0.98, *n.s.*), entrepreneurship (W = 0.98, *n.s.*), perseverance (W = 0.99, *n.s.*), imagination (W = 0.97, *n.s.*), and co-operation (W = 0.97, *n.s.*). These results indicated that only cooperativity factor did not show normality. We applied parametric analysis because the sample size was close to 100 and there was one factor that did not show normality.

Table 3. Results of the Normality Test of the Computational Thinking Scale and Creative Attitudes

		W	
Computational Thinking Scale	*creativity*	0.99	
	algorithmic thinking	0.98	
	cooperativity	0.95	**
	critical thinking	0.98	
	problem solving	0.99	
Creative Attitudes	*flexibility*	0.99	
	analytical problem solving	0.98	
	entrepreneurship	0.98	
	perseverance	0.99	
	imagination	0.97	
	cooperation	0.97	
**p<.01		(n = 93)	

3.3. Correlation between Computational Thinking Scale and Creative Attitudes

Pearson's correlation coefficient was used to calculate the correlation coefficient, as shown in Table 4.

We focused only on the relationship between CTS and creative attitudes, and items with correlation coefficients greater than 0.40. The results showed significant associations between creativity and the following: flexibility, analytical problem solving, entrepreneurship, and perseverance. Moreover, we determined the significant associations between algorithmic thinking and the following: flexibility, analytical problem solving, entrepreneurship, and perseverance. We found the same results for cooperativity and co-operation. Moreover, there were significant associations between critical thinking and the following: flexibility, analytical problem solving, entrepreneurship, and perseverance. The correlation coefficients between problem solving and all factors of creative attitudes were less than 0.40.

4. DISCUSSION

The creativity of CTS was related to creative attitudes of flexibility, analytical problem solving, entrepreneurship, and perseverance. Previous research showed a relationship between creativity and computational thinking.

Flexibility and analytical problem solving, entrepreneurship and perseverance correlated with creativity, critical thinking, and algorithmic thinking. This suggests that it is adequate to focus on these relationships in order to design activities that enhance creative attitudes and computational thinking. Similarly, focusing on the relationship between entrepreneurship and perseverance may enhance creative attitudes and computational thinking.

However, imagination did not correlate with all factors of creative attitudes, and co-operation correlated with cooperativity only. Therefore, activities that aim to increase the imagination and co-operation of creative attitudes to enhance computational thinking may not be efficient.

The participants of this study were enrolled in a course that dealt with game development. Many of them were students aiming to create new games. In addition, they had numerous programming classes, and typically performed programming using Python and Unity.

When creating a game of a certain scale, it is necessary to focus on the relationship between the whole and its parts, such as how to create modules, consider the overall design, and proceed with development in a structured manner. This suggests that algorithmic thinking is related to an analytical attitude toward problems and a participant's attitude.

Moreover, problem solving in CTS is making several choices or aiming to solve problems collaboratively. With the COVID-19 pandemic, tasks are often performed individually and there is limited development within teams, thus, it is assumed that there is no relationship between problem solving and several factors of creative attitudes. The same applies to cooperativity in CTS.

Thus, it is unlikely that a unique curriculum focusing on each CTS factor can be developed to enhance CTS's creativity, critical and algorithmic thinking. Moreover, to improve the cooperativity and problem solving of CTS, it is necessary to consider an individual's curriculum. Fostering cooperativity and problem solving independently through

Table 4. Correlation between Computational Thinking Scale and Creative Attitudes

	creativity	*algorithmic thinking*	*cooperativity*	*critical thinking*	*problem solving*	*flexibility*	*analytical problem solving*	*entrepreneurship*	*perseverance*	*imagination*	*cooperation*
creativity	1.00										
algorithmic thinking	**0.55****	1.00									
cooperativity	0.20	0.08	1.00								
critical thinking	**0.60****	**0.63****	0.07	1.00							
problem solving	0.17	0.20	0.22*	0.15	1.00						
flexibility	**0.61****	**0.62****	0.22*	**0.69****	0.22*	1.00					
analytical problem solving	**0.56****	**0.58****	0.13	**0.69****	0.22*	**0.67****	1.00				
entrepreneurship	**0.59****	**0.44****	0.18	**0.55****	0.05	**0.55****	**0.68****	1.00			
perseverance	**0.60****	**0.46****	0.21*	**0.63****	0.09	**0.57****	**0.69****	**0.59****	1.00		
imagination	0.28**	0.37**	0.00	0.24*	-0.21*	**0.41****	0.38**	**0.53****	0.32**	1.00	
cooperation	0.07	-0.02	**0.46****	-0.05	0.24	0.10	0.07	0.14	0.13	-0.12	1.00

**p< .01, *p< .05 (n = 93)

activities that enhance creative attitudes and computational thinking is necessitated.

Table 4 shows that the items other than co-operation were related to each other in each factor of creative attitudes. In developing subjects, it is essential to create subjects and practices that foster computational thinking and creative attitudes in a well-balanced manner, not by correlating items.

5. SUMMARY AND FUTURE WORK

This study explored the relationship between CTS and the creative attitudes of university students to obtain basic knowledge for designing classes to enhance computational thinking and creativity. Although the relationship between computational thinking and creativity was determined, we focused on creative attitudes and creativity as readiness and clarified that the relationship between computational thinking and creative attitudes provides essential knowledge for future lesson design. Results in this study demonstrated a correlation, however participants of the survey were students specializing in game development and programming. Therefore such a cohort would likely have the requisite skills and attitudes surveyed.

However, there are some limitations which should be addressed in future studies. First, there is a need to expand the number of survey participants. It is assumed that university students from other departments have different tendencies toward CTS than those surveyed in this study.Consequently, it is necessary to survey various students to understand the relationship between CTS and creative attitudes in more detail. In addition, the number of participants in this study were 93, which was not large. A larger sample is required to examine the validity and reliability of these factors.

Second, correlations with scores of other creativity tests, such as the S-A is essential. The creative attitudes psychological scale is sufficient for understanding creative tendencies, however students' creativity is not easily understood. Therefore, it is necessary to use a creativity test to examine the relationship between CTS and creativity.

In the future, these problems should be resolved, and practices should be developed based on the survey results. Practices that enhance computational thinking and creativity should be implemented.

6. ACKNOWLEDGMENTS

This work was partially supported by JSPS KAKENHI (Grant Number 21K13644), and the Masason Foundation. We are grateful to Associate Prof. Tetsuya Bando (Naruto University of Education), and Aya Morozawa (Kanto Gakuin University).

7. REFERENCES

Bando, T. & Motozawa, A. (2021) The Relationship between Computational Thinking and Grit among University Students, *Journal of the Japan Society of Technology Education*, 63(1), 23-29. (in Japanese)

Brennan, K., & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking, *Annual American Educational Research Association Meeting*. Retrieved January 5, 2022, from http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Computing at School. (2014). Secondary Computing Guidance: Computing in the National Curriculum: A Guide for Secondary Teachers. Retrieved January 5, 2022, from https://community.computingatschool.org.uk/files/3383/original.pdf

Computing at School. (2015). *Computational Thinking Teacher Resources*. Retrieved January 5, 2022, from https://community.computingatschool.org.uk/files/6890/original.pdf

Computing at School. (2015a). *Computing Progression Pathways.* Retrieved January 5, 2022, from https://community.computingatschool.org.uk/files/5094/original.pdf

Doleck, T., Bazelais, P., Lemay, D., Saxena, A., & Basnet, R. B. (2017). Algorithmic Thinking, Cooperativity, Creativity, Critical Thinking, and Problem Solving: Exploring the Relationship between Computational Thinking Skills and Academic Performance. *Journal of Computers in Education*, *4*(4), 355-369. https://doi.org/10.1007/s40692-017-0090-9

Fagerlund, J., Häkkinen, P, Vesisenaho, M., & Viiri, J. (2020). Computational Thinking in Programming with Scratch in Primary Schools: A Systematic Review. *Computer Applications in Engineering Education*, 29(1) 1-17. https://doi.org/10.1002/cae.22255

Gov.UK: Department for Education (2013). National Curriculum in England: Computing Programmes of Study. Retrieved January 5, 2022, from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

Grover, S., & Pea, R. D. (2015) "Systems of Assessments" for Deeper Learning of Computational Thinking in K-12, *Conference:Annual Meeting of the American Educational Research Association*.

Harvard University. (n.d.). *Computational Thinking with Scratch.* Retrieved January 5, 2022, from http://scratched.gse.harvard.edu/ct/index.html

Hershkovitz, A., Sitman, R., Israel-Fishelson, R., Eguíluz, A., Garaizar, P., & Guenaga, M. (2019). Creativity in the Acquisition of Computational Thinking. *Interactive Learning Environments*, *27*(5-6), 628-644. https://doi.org/10.1080/10494820.2019.1610451

International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA). (2011) *Operational Definition of Computational Thinking for K-12 Education*. Retrieved January 5, 2022, from https://cdn.iste.org/www-root/Computational_Thinking_Operational_Definition_ISTE.pdf

International Society for Technology in Education (ISTE). (n.d.) *Exploring Computational Thinking Resource Repositories*. Retrieved January 5, 2021, from

https://learn.iste.org/d2l/lor/search/search_results.d2l?ou=6606&lrepos=1006

Korkmaz, Ö., Cakir, R., & Özden, M. Y. (2017). A Validity and Reliability Study of the Computational Thinking Scales (CTS). *Computers in Human Behavior*, *72*, 558-569.

Kurzweil, R. (2005). *The Singularity Is Near: When Humans Transcend Biology*. New York: Viking.

Papert, S. (1980) *Mindstorms: Children, Computers, and Powerful Ideas.* New York: Basic Books.

Rotem, I-F, Hershkovitz, A., Eguíluz, A., Garaizar, P., & Guenaga, M. (2020) The Associations Between Computational Thinking and Creativity: The Role of Personal Characteristics, *Journal of Educational Computing Research*, *58*(1), 1415-1447. https://doi.org/10.1177/0735633120940954

Schank, R., & Childers, P. (1988) *The Creative Attitude.* New York: Macmillan Publishing Company.

Selby, C., & Woollard, J. (2013). *Computational Thinking: the Developing Definition*. Retrieved January 5, 2022, from http://eprints.soton.ac.uk/356481

Shigemasu, K., Yokoyama, A., Stern, S., & Komazaki, H. (1993) Comparison of Creative Attitudes between American and Japanese Students: A Factor Analytical Study, *The Journal of Japanese Journal of Psychology*, *64*(3), 181-190. (in Japanese)

Shute, V. J., Sun, C., & Jodi Asbell-Clarke b. (2017) Demystifying Computational Thinking, *Educational Research Review*, *22*, 142-158.

Wing, M. J. (2014) *Computational Thinking Benefits Society*. Social Issues in Computing, Retrieved January 5, 2022, from http://socialissues.cs.toronto.edu/2014/01/computational-thinking

Appendix

We denote the computational thinking scale (CTS) and creative attitudes in Tables 5 and 6.

Table 5. The Factors and Items of the Computational Thinking Scale

Factor 1: Creativity	
1	I like the people who are sure of most of their decisions.
2	I like the people who are realistic and neutral.
3	I believe that I can solve most of the problems I face if I have sufficient amount of time and if I show effort.
4	I have a belief that I can solve the problems possible to occur when I encounter with a new situation.
5	I trust that I can apply the plan while making it to solve a problem of mine.
6	Dreaming causes my most important projects to come to light.
7	I trust my intuitions and feelings of “trueness” and “wrongness” when I approach the solution of a problem.
8	When I encounter with a problem, I stop before proceeding to another subject and think over that problem.
Factor 2: Algorithmic Thinking	
9	I can immediately establish the equity that will give the solution of a problem.
10	I think that I have a special interest in the mathematical processes.
11	I think that I learn better the instructions made with the help of mathematical symbols and concepts.
12	I believe that I can easily catch the relation between the figures.
13	I can mathematically express the solution ways of the problems I face in the daily life.
14	I can digitize a mathematical problem expressed verbally.
Factor 3: Cooperativity	
15	I like experiencing cooperative learning together with my group friends.
16	In the cooperative learning, I think that I attain/will attain more successful results because I am work.
17	I like solving problems related to group project together with my friends in cooperative learning.
18	More ideas occur in cooperative learning.
Factor 4: Critical Thinking	
19	I am good at preparing regular plans regarding the solution of the complex problems.
20	It is fun to try to solve the complex problems.
21	I am willing to learn challenging things.
22	I am proud of being able to think with a great precision.
23	I make use of a systematic method while comparing the options at my hand and while reaching a decision.
Factor 5: Problem Solving	
24*	I have problems in the demonstration of the solution of a problem in my mind.
25*	I have problems in the issue of where and how I should use the variables such as X and Y in the solution.
26*	I cannot apply the solution ways I plan respectively and gradually.
27*	I cannot produce so many options while thinking of the possible solution ways regarding a problem.
28*	I cannot develop my own ideas in the environment of cooperative learning.
29*	It tires me to try to learn something together with my group friends in cooperative learning.

*invert scale

Table 6. The Factors and Items of the Creative Attitudes

Factor 1: Flexibility	
1*	I am not good at making analogies.
2	I enjoy talking about a wide variety of topics.
3	I am good at finding common characteristics among dissimilar things.
4	Even when I encounter a difficult problem, I can usually find a solution.
5	I am knowledgeable about many different subjects.
6	Other people often say that I think of ideas that are different from the ideas of others.
7	I can think of many related ideas.
8	I think of many different ideas at the same time.
9	I enjoy participating in intense discussions.
10	I can easily think of alternatives when I have difficulty in solving a problem.
11	Other people often ask me how to solve a problem.
12	I can think of several different ways to solve a problem at the same time.
13	I usually view a situation from several different perspectives.
14	I often have several different views about a phenomenon.
15	I can easily divert myself.
16	Other people consider me to be unusual.
17	I am good at understanding things.
18	I always consider the possibility that other people's ideas may be wrong.
19*	I dislike unusual people.
20	I am able to understand other people's feelings.
21	I often wonder what the world will be like in the future.
22	It is easy for me to ask others for help.
23	I prefer to solve problems in my own way.
24*	It is difficult for me to voice opinions that are different from the majority.
Factor 2: Analytical Problem Solving	
25	I think about the structure of a problem before I begin to solve it.
26	I often consider the fundamental basis of things.
27*	It is difficult for me to understand structure of a problem.
28	Before starting to do something. I usually think about the process.
29	I can easily divide a problem into several subproblems.
30	After I solve a problem, I continue to try to find more beautiful solutions.
31	When I think of a new idea, I also think of how to implement it at the same time.
32	I believe in my ability to understand the fundamental nature of things.
33	I enjoy making detailed observations.
34	I am not interested in ordinary things.
35*	I tend to make judgements intuitively, rather than logically.
Factor 3: Entrepreneurship	
36	I want to create new and beautiful things that have never been made before.
37*	I dislike things that are new and unusual.
38	I am very curious.
39	I am interested in learning about things that are not related to what I am doing.
40	I want to create beautiful things.
41	I am interested in many different things.
42	I want to create things that are better than those made by other people.
43	I like to create new things that make life more convenient.
44	I have a good understanding of what makes art and music beautiful.
45	I am good at making things.
46	I like to take things apart.
47	I start working on a new idea, even when I don't yet know how to do it.
Factor 4: Perseverance	
48	I have strong opinions that usually do not change.
49	I do not like to leave things unfinished.
50	During a discussion, I usually do not change my opinions.
51	I usually accomplish what I set out to do.
52	I tend to stick to my old ideas.
53	When I concentrate, I am not aware of things around me.
54	I do not give up easily.
55	I find it difficult to understand things that are inconsistent and illogical.
56	I become frustrated when I cannot solve a problem.
57	It is easy for me to keep working on the same task for a long time.
58	I am confident that if I think I will succeed, I will succeed.
Factor 5: Imagination	
59	I daydream often.
60	I can easily imagine things that do not exist.
61	Thinking about something new makes me happy.
62	Rather than concentration on one task, I prefer to move back and forth from one task to another.
63	I think that I am very different from other people.
64	I have strong emotional feelings several times a day.
Factor 6: Cooperation	
65	I often consider the group consensus when I work with others to solve a problem.
66	The approval of people in authority is important to me.
67	I often work in cooperation with others.
68	I usually consult others when I don't know how to solve a problem.
69	When I see someone who is having difficulty, I usually try to help them.
70	I want to try to do things that are good for both myself and society.
71*	I see unusual aspects of common occurrences.
72*	I am reluctant to share my ideas with others.
73*	I think I will be more creative when I work by myself, rather than when I work with others.
74*	I care about what others think about me.

*invert scale

DOI: 10.34641/ctestem.2022.472

A Review of Reviews on Computational Thinking Assessment in Higher Education

Xiaoling Zhang, Marcus Specht
Delft University of Technology, Netherlands
X.Zhang-14@tudelft.nl, M.M.Specht@tudelft.nl

ABSTRACT
There is an urgent need for educating the next generation of learners with digital tools and making use of digital practices and skills. Education on computational thinking (CT) is widespread around the world with a dominant focus on K-12. Recently also higher education has come more to the focus of CTE. However, most of the work on CT in higher education has been focused on teaching and learning programming while less attention has been paid to the underlying skills and competences of CT in different domains. In this article 11 reviews were analyzed to identify constructs being assessed, methods and their characteristics for the delivery of assessment and the context in which the assessment were conducted. The findings indicate that there is certain consensus in the field on what constructs to measure. Last but not least, it was determined from our study that there are often no standards or principles followed for the design of assessment.

KEYWORDS
Computational Thinking, Assessment, Higher Education, Literature Review

1. INTRODUCTION
According to Denning (2016), Computational Thinking (CT) is skillset that human beings utilized for problem-solving regardless of the rapid change of technology throughout history. Additionally, the importance of CT for modern citizens is stressed in Royal Society (2012) in UK and Royal Netherlands Academy of Arts and Sciences (2013) since CT is regarded as imperative for enabling people to better work and live in a digital environment.

Though the long historical usage of CT skills was highlighted by Denning (2016), research in the field of CT education is still in its early age. Computational thinking was first mentioned by Papert (1980) in his book and then promoted by Wing (2006)'s viewpoint delivered through Communications of the ACM, described as an imperative skill for everyone just like 3R (reading, writing, arithmetic). Since then, researchers, practitioners and policymakers who are proponents and critics of this topic started to explore and study teaching, learning and assessment of CT by examining different dimensions of the topic across all education levels with more attention to K-12 education. Dimensions being studied include but are not limited to the definition of CT (Lyon & J. Magana, 2020; Shute et al., 2017; Tang et al., 2020), the integration of CT to the current curriculum (García-Peñalvo, 2017; Henderson et al., 2007; Leathem et al., 2019), the interventions used for CT teaching (Constantinou & Ioannou, n.d.; Ezeamuzie & Leung, 2021; Taslibeyaz et al., 2020), the tools developed for CT teaching and learning (Ambrósio et al., 2015; Angeli & Giannakos, 2020), or the assessment of CT (Y. Li et al., 2020; Rom An-Gonz Alez et al., 2016; Sondakh et al., 2020). Often it still remains unclear what CT is, how they are operationalized in educational activities, what distinguishes it from other kinds of thinking skills and how it can be incorporated with other subject domains (Specht, 2020).

Irrespective of the controversies and ambiguity mentioned above, a considerable amount of knowledge has been accumulated in this field. Theoretical frameworks have been established over the years to facilitate CT understanding and promotion, such as Brennan and Resnick's (2012) three-dimensional framework, Weintrop's (2016) taxonomy of CT in mathematics and science, Grover and Pea's (2018) competency framework, which are seemingly the most adopted ones in the literature. Tools such as Alice, Scratch, Bebras (Tang et al., 2020), have been developed for teaching, learning and assessment of CT in different contexts (Cutumisu et al., 2019). Curriculums have been developed for teaching CT in different contexts (Hsu et al., 2018). It is noteworthy that though CT has also been interpreted as a model of thinking essential for everyone which can be applied to not only a broad range of domains such as engineering and mathematics, but also daily life scenarios relevant to problem solving (Angeli & Giannakos, 2020; X. Li & Gu, 2020; Tedre & Denning, 2016; Wing, 2006), CT has been mostly linked to Computing Science and programming and more contributions are made in the context of K-12 than in higher education (Cutumisu et al., 2019).

In the process of CT education, assessment is a core component for ensuring learning outcomes. As Van de Vleuten et al. (2011) concluded, the determining factors for the quality of an assessment program and the quality of assessment consist of the types of constructs being assessed, the method used for collecting and collating of information, the role of human judgement and the psychometric methods which requires further investigation. Some of those factors have been examined in several reviews on CT studies in higher education, nonetheless, there is no work providing a holistic view on those factors affecting the quality of assessment up to our knowledge. Thus, in this work, drawing on the conclusion of van de Vleuten (2011), we aim to systematically investigate and synthesis existing knowledge within the following dimensions in terms of CT assessment in higher education: the types of constructs being assessed, methods used for collection and collation of the information, the role of human judgement and the psychometric methods and the assessment context (an extra dimension which lay the background for conduction of assessment) and some additional characteristics of assessment methodologies.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

The method we applied is a systematic umbrella review of CT assessment in higher education to provide an holistic view on the 5 dimensions crucial for assessment mentioned in the previous paragraph by answering the following research questions (RQs):

RQ1 What are the characteristics of the included reviews, such as year of publication, country of the work, type of publication, and the methodological features such as type of study, principles, methodology followed for the study and tools that are used?

RQ2 What knowledge that can help suggest developing a high quality of assessment program has been gathered in existing studies regarding assessment of CT in higher education?

- **RQ2.1:** Assessment objects/constructs: which components were examined as assessment constructs?
- **RQ2.2:** Assessment methodology: What perspectives of assessment methodology have been examined?
- **RQ2.3:** Assessment context: What is the assessment context in which CT has been measured?

2. METHODOLOGY

The study adopted the systematic process depicted by Jesson et al. (2011) for gathering the literature to be used as the data set. The major steps followed were (1) identify scope and research question, (2) plan the review and document protocol, (3) develop inclusion and exclusion criteria, (4) search and screen the studies, (5) data extraction and synthesis. The first two steps were performed through narrative analysis on literature and with assistance of an expert and PRISMA is adopted as the plan for primary steps of the review and the others are documented in an excel file. The quality of the studies included in this review was examined through discussions between the authors where necessary. The rest of the steps will be reported in detail in the following subsections. The principal results for the key steps were recorded in the PRISMA flowchart and are shown in Figure 1.

Identification: Number of records through database searching (Scopus = 212, WoS = 143) = *355 + Google Scholar (first ten pages most relevant results)*
Screening: Number of records after duplicate removal = Number of records for screening = *298 (Scopus + WoS)*
Eligibility Checking: Number of full-text articles assessed for eligibility = *129 (Scopus + WoS)*
Included: Number of studies included for analysis = 7 (Scopus + WoS) + 3 (Google Scholar) = *11 in total*

Figure 1. PRISMA Flowchart with Results

3. RESULTS AND ANALYSIS

3.1. General Characteristics of the Included Studies

Overall, eleven studies examined CT assessment in higher education have been included in this analysis. In terms of bibliographical characteristics of the included studies were published within the last five years and no studies have been found before the year of 2016, indicating an increase in the attention to this topic. Over those years, the effort into CT can be found worldwide, with the United States, Turkey and Canada the most active ones. The contributions of those countries in the last years were published as journal articles, conference papers, and book chapters in journals such as Informatics in Education, conferences such as Frontiers in Education Conference and books such as Research on E-Learning and ICT in Education, respectively.

Regarding the methodological characteristics over the included studies the type of study that the review belongs to, it can be observed that most of the included studies are systematic review, followed by scoping review, narrative review and systematic mapping study. In addition to that, besides the work of Vinu (2021), the rest of the other studies referred to existing methods or guidelines for conducting a review.

Among all reviews studied, only Lu (2022)'s work fully focused on investigating empirical evidence of CT assessment in higher education. Eight included reviews examined objects being assessed and characteristics of the objects in their selected studies regarding the skills and competencies and the underlying constructs. Of those studies left, two examined the definition of CT with which the assessed constructs can be deduced by applying constructive alignment theory (Biggs, 1996). In terms of assessment methodology, except for De Jong and Jeuring (2020)'s work, all other studies examined perspectives relevant to the delivery of the assessment, namely - instrument developed for assessment and its characteristics, tools used for assessment and its characteristics, the categorization of assessment methods, the quality indicators for the assessment methods. The context in which the assessment is conducted is examined by all included reviews within the following perspectives: educational setting, education level and academic domain and studies. Detailed examination and result analysis are presented in the following subsections.

3.2. Characteristics of Assessment Objects/Constructs

Cutumisu et al. (2019) and Lu et al. (2022) outlined the assessment constructs by mapping the assessed CT skills to Brennan and Resnick (2012)'s three-dimensional framework and a hybrid framework inferred from Brennan and Resnick's (2012) three-dimensional framework, Weintrop et al. (2016) framework of CT for mathematics and science classrooms, and Grover and Pea (2018)'s two-dimensional framework, respectively. Though the framework of Brennan and Resnick (2012) adopted by Cutumisu et al. (2019) and the hybrid framework adopted by Lu et al. (2022) both depicted CT competency in a three-dimensional framework inclusive of CT concepts, practices and perspectives, the latter was claimed to be more generic

and independent of specific subjects which also allows a broader coverage of CT skills and dimensions.

The other five studies presented only the overarching categories of assessed constructs (De Jong & Jeuring, 2020; Taslibeyaz et al., 2020) that provide high-level categorization of constructs being assessed without revealing the constructs itself; or the constructs being assessed in studies (De Araujo et al., 2016; Poulakis & Politis, 2021) or both the constructs and its overarching categories (Hasesk et al., 2019).

It is noticeable that some categories are named almost the same, such as CT skills versus CT skills / ability and attitudes towards CT versus attitude-motivation. However, it is considered improper, by the authors, to merge them at the current level of investigation with insufficient information on its meaning. Thus, the categories of constructs and the constructs are regarded as distinct elements unless they are proven to be identical. The results also show that Taslibeyaz (2020)'s and De Jong and Jeuring (2020)'s works identified six distinct categories of constructs including attitude towards CT, attitude-motivation, CT knowledge, CT skills, problem-solving skills, programming skills while De Araujo (2016), Haseski (2019), and Poulakis (2021) identified five categories of CT construct consist of affective achievements towards CT, cognitive achievements towards CT, CT skills / abilities, CT concepts, CT patterns with enumeration of the underlying constructs in their reports.

Table 1 presents the constructs which appeared in at least 3 reviews while 120 unique constructs were identified from all reviews since it can be too long to present it here. Additionally, the constructs were categorized according to the hybrid framework in Lu et al. (2022)'s work. Example definitions of these constructs from the work are provided in the table when it is accessible according to the hybrid framework.

Table 1. Assessed Constructs Identified from Reviews.

Category	**Constructs & Frequency (f)**	**Definition**
CT Concepts	Algorithm / algorithmic thinking / algorithm skills (f =5)	The skills involved in developing an algorithm which is precise with step-by-step procedures to solve a problem. (Grover & Pea, 2018).
	Data / data analysis / data collection, data analysis / data representation (f=5)	Including storing, retrieving, updating values as well as analyzing, and visualizing data (Brennan & Resnick, 2012; Weintrop et al., 2016).
	Automation / automating solutions (f=4)	A key component of computational thinking, for computer science as well as computing in other domains that aims at a solution to be executed by a machine (Grover & Pea, 2018).
	Logic / logic and logical thinking (f=4)	Logical thinking involves analyzing situations to make a decision or reach a conclusion about a situation (Grover & Pea, 2018).
	Critical thinking (f=3)	Not found.
	Evaluation (f=3)	Solutions to problems are evaluated for accuracy and correctness with respect to the desired result or goal(Grover & Pea, 2018).
	Pattern / pattern recognition (f=3)	Pattern recognition in CT could result in a definition of a generalizable solution which can utilize automation in computing for dealing with a generic situation (Grover & Pea, 2018).
	Synchronization / synchronize (f=3)	Not found.
CT Practices	Abstraction (f=5)	Abstraction is 'information hiding'. Through 'black-box'-ing details, one can focus only on the input and output and provides a way of simplifying and managing complexity (Grover & Pea, 2018; Weintrop et al., 2016).
	Problem-solving (f=4)	Not found.
	Modularity / modularizing / modelling (f=3)	Building something large by putting together collections of smaller parts, is an important practice for all design and problem solving (Brennan & Resnick, 2012).
	Testing / testing and debugging (f=3)	Practices that are relevant to dealing with – and anticipating – problems include identifying the issue, systematically evaluating the system to isolate the error and reproducing the problem so that potential solutions can be tested reliably (Grover & Pea, 2018; Weintrop et al., 2016).
CT Perspectives	Creativity and creation (f=4)	Creativity as a CT practice acts on two levels – it aims to encourage

		out-of-the-box thinking and alternative approaches to solving problems; and it aims to encourage the creation of computational artefacts as a form of creative expression (Grover & Pea, 2018).
	Collaboration and cooperation (f=3)	Perspectives about working with CT skills in a collaborative or cooperative format (Grover & Pea, 2018).

3.3. Characteristics of Assessment Methodology

There are 10 out of 11 studies investigating the topic of assessment methodology. As shown in table 2, four of them discussed the types of assessment methods emerged from their investigation.

Table 2. Assessment Methods.

Types of Assessment Methods	Reference
Block-based assessments, knowledge/skill written tests, self-reported scales/survey, robotics/game-based assessments (tangible tasks), combinations	(Cutumisu et al., 2019)
Block-based assessments, knowledge/skill written tests, self-reported scales/survey, text-based programming assessment, course academic achievements of CS courses, interviews and observations, combinations	(Lu et al., 2022)
Interviews, Assignment/course grades, survey/questionnaire, knowledge/skill tests, artefacts (classroom/students), problems external to class, combinations	(Lyon & J. Magana, 2020)
Using specific programming environments, using CT assessment criteria and/or psychometric tools, using multiple forms of assessment	(Poulakis & Politis, 2021)

Lu et al. (2022) and Cutumisu et al. (2019) both identified the following types of assessment in their work: **1) block-based assessments** - solving programming problems without taking into account syntax by using programming blocks in block-based programming environments such as Scratch; **2) skill written tests** - using generic forms for assessment such as constructed response questions or multiple-choice questions to assess CT skills, e.g. Computational Thinking Knowledge test (CT Knowledge test); **3) self-reported scales / survey** - mostly concerned with assessment of CT perspectives which includes inter- and intra-personal skills such as communication, collaboration, or questioning, for example, Computational Thinking Scales (CTS) is a questionnaire that measures five factors including communication, critical thinking, problem-solving, creative thinking and algorithmic thinking. In addition to that, Cutumisu et al. (2019) also identified **robotics/game-based assessments** as a unique category with which indicating the assessments that are based on robotic tangible tasks or artefacts produced in game-based assessments such as AgentSheets. Lu et al. (2022) identified another three categories compared with categories of Cutumisu et al. (2019)'s work, being: **text-based programming assessments** - using text-based programming tasks to assess students' CT competency, for example, a Python programming task; **interviews and observations** - commonly used for studying practices of incorporating CT into traditional classrooms; **course academic achievement** - academic performance in coursework including students achievement in quizzes, exam, projects and assignments.

3.4. Characteristics of Assessment Context

All studied reviews contain information about assessment context. A summary of major aspects its corresponding references is presented in table 3.

Table 3. Assessment Context

Aspects	Description	Reference
Academic Domain	Concerned with investigation into academic disciplines, program of studies or subject matter for the assessed group of users.	(Cutumisu et al., 2019; De Jong & Jeuring, 2020; Ezeamuzie & Leung, 2021; Lu et al., 2022; Lyon & J. Magana, 2020; Tang et al., 2020; Taslibeyaz et al., 2020)
Education Level	Concerned with the level of education for the assessed group of users.	All included reviews
Educational Setting	Concerned with the type of educational activities that the assessed group of users were involved in.	(Cutumisu et al., 2019; De Araujo et al., 2016; Ezeamuzie & Leung, 2021; Lu et al., 2022; Tang et al., 2020)
Intervention	Concerned with the actions taken for the development of skills and / or their corresponding characteristics.	(Cutumisu et al., 2019; De Jong & Jeuring, 2020; Ezeamuzie & Leung, 2021; Lyon & J. Magana, 2020; Taslibeyaz et al., 2020; Vinu Varghese & Renumol, 2021)

For academic domain, besides De Jong and Jeuring (2020) presented a list of academic disciplines, distinguishing the academic background according to the relevance of its study program to Computer Science (CS), Science, Technology, Engineering and Technology (STEM), and Programming Education is found a phenomenon across the studies (De Jong & Jeuring, 2020; Ezeamuzie & Leung, 2021; Tang et al., 2020; Taslibeyaz et al., 2020).

In terms of education level, all reviews included for analysis examined it. However, results were presented differently

varying from listing the exact grade level of the examined studies to showing the distribution of grade level in categories with descriptive text. Even with the studies of Lu et al. (2022), Lyon (2020) and De Jong and Jeuring (2021) which delimited their studies in higher education, Lu et al. (2022) presented the exact grade level of the examined studies in a table while the other two regard undergraduate itself as a category.

4. CONCLUSION AND FUTURE WORK

To comprehensively study existing knowledge on CT assessment in higher education, this study systematically reviewed eleven reviews that either fully focused on review of CT assessment in higher education or include CT assessment in higher education as a composition of all investigation dimensions.

In terms of the bibliographical and methodological characteristics of the included studies (for answering RQ1), it was determined that there is a worldwide increase in attention to explore knowledge on the topic of assessment of CT higher education from different dimensions from various perspectives.

To gain a comprehensive view about major components in an assessment and to answer RQ2, this work identified, regarding CT assessment in the context of higher education, constructs being assessed, the characteristics of methodology for assessment and assessment context. Only one of the three works which examined CT in higher education specifically studied assessment (De Jong & Jeuring, 2020; Lu et al., 2022; Lyon & J. Magana, 2020).

First of all, regarding the constructs being assessed in assessments, this work identified more than 100 unique constructs. While some work clustered the constructs from included studies, only with the work of Cutumisu et al. (2019) and Lu et al. (2022) identified constructs by drawing on Brennan and Resnick's CT framework and the hybrid framework consisting of Brennan and Resnick's framework, Grover and Pea's framework and Weintrop (2016)'s framework, respectively. None of the studies examined if certain constructs or constructs types appeared more often and considered more appropriate to be assessed at a certain educational level.

Moreover, assessment methods were categorized differently in the four reviews in which the methods are grouped and presented. It is recognized that whether the method concerns with programming a major distinguishing factor. Meanwhile, combined use of different assessment methods were positively promoted and suggested from the results and the text of those four reviews.

Furthermore, with regard to assessment context, this study identified four major dimensions that provide information about academic background: academic domain, education level, educational setting and intervention. Results show that there is an increased number of studies bringing CT into various disciplines, with more attention to non-CS majors in recent years. Most studies are conducted in a formal educational setting and assessments are mostly conducted with entry-level or lower-level students in higher education which is integrated in a course or curriculum.

While CT education is a part of education, existing studies apply no assessment framework or reason about the design of the assessment. We argue that design of assessment, especially assessment for learning, plays a critical role in assisting students in learning skills. This study provides an overview of potential factors that need to be considered, according to the evidence of existing research, when designing assessment for assessing CT skills.

5. AUTHOR STATEMENT AND DATA ACCESS STATEMENT

5.1. Author Statement

Xiaoling Zhang: Conceptualization, Methodology, Data Collection, Analysis, Writing - Original Draft, Writing – Review & Edit, Visualization, Resources

Marcus Specht: Conceptualization, Methodology, Writing – Review & Edit

This work is a part of a PhD project funded by Center for Education and Learning at Leiden-Erasmus-Delft Universities (LDE-CEL).

5.2. Data Access Statement

All papers used for analysis in this work can be approached via the following three indexing databases: SCOPUS, Web of Science and Google Scholar. Collated data that is used for reporting the results are accessible upon request to authors..

6. REFERENCES

Ambrósio, A. P., Xavier, C., & Georges, F. (2015). Digital ink for cognitive assessment of computational thinking. *Proceedings - Frontiers in Education Conference, FIE*, *2015-Febru*(February). https://doi.org/10.1109/FIE.2014.7044237

Angeli, C., & Giannakos, M. (2020). Computational thinking education: Issues and challenges. *Computers in Human Behavior*, *105*. https://doi.org/10.1016/J.CHB.2019.106185

Biggs, J. (1996). Enhancing teaching through constructive alignment. *Higher Education*, *32*(3), 347–364. https://doi.org/10.1007/BF00138871

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Annual American Educational Research Association Meeting, Vancouver, BC, Canada*, *2012*, 1–25. http://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf

Constantinou, V., & Ioannou, A. (n.d.). *Development of Computational Thinking Skills through Educational Robotics*.

Cutumisu, M., Adams, C., & Lu, C. (2019). A Scoping Review of Empirical Research on Recent Computational Thinking Assessments. *Journal of Science Education and Technology*, *28*(6), 651–676. https://doi.org/10.1007/s10956-019-09799-3

De Araujo, A. L. S. O., Andrade, W. L., & Serey Guerrero, D. D. (2016). A systematic mapping study on assessing computational thinking abilities.

Proceedings - Frontiers in Education Conference, FIE, *2016-Novem*. https://doi.org/10.1109/FIE.2016.7757678

De Jong, I., & Jeuring, J. (2020). Computational Thinking Interventions in Higher Education: A Scoping Literature Review of Interventions Used to Teach Computational Thinking. *ACM International Conference Proceeding Series*, *10*(20). https://doi.org/10.1145/3428029.3428055

Ezeamuzie, N. O., & Leung, J. S. C. C. (2021). Computational Thinking Through an Empirical Lens: A Systematic Review of Literature. *Journal of Educational Computing Research*, 073563312110331. https://doi.org/10.1177/07356331211033158

García-Peñalvo, F. J. (2017). Computational thinking issues. *ACM International Conference Proceeding Series*, *Part F1322*. https://doi.org/10.1145/3144826.3145349

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer Science Education: Perspectives on Teaching and Learning in School*, *19*(December), 19–37.

Hasesk, H. I., Ilic, U., Haseskİ, H. İ., & İlİc, U. (2019). An Investigation of the Data Collection Instruments Developed to Measure Computational Thinking. *Informatics in Education*, *18*(2), 297–319. https://doi.org/10.15388/infedu.2019.14

Henderson, P. B., Cortina, T. J., & Wing, J. M. (2007). Computational thinking. *ACM SIGCSE Bulletin*. https://doi.org/10.1145/1227504.1227378

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers and Education*, *126*, 296–310. https://doi.org/10.1016/j.compedu.2018.07.004

Leathem, T., Hillesheim, C., Coley, A., & McGregor, S. (2019). Student and teacher perspectives on a multi-disciplinary collaborative pedagogy in architecture and construction. *Higher Education, Skills and Work-Based Learning*, *9*(1), 121–132. https://doi.org/10.1108/HESWBL-03-2018-0026

Li, X., & Gu, C. (2020). Teaching reform of programming basic course based on SPOC blended teaching method. *15th International Conference on Computer Science and Education, ICCSE 2020*, 411–415. https://doi.org/10.1109/ICCSE49874.2020.9201802

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). On Computational Thinking and STEM Education. *Journal for STEM Education Research*, *3*(2), 147–166. https://doi.org/10.1007/s41979-020-00044-w

Lu, C., Macdonald, R., Odell, B., Kokhan, V., Demmans Epp, C., & Cutumisu, M. (2022). A scoping review of computational thinking assessments in higher education. *Journal of Computing in Higher Education*, *0123456789*. https://doi.org/10.1007/s12528-021-09305-y

Lyon, J. A., & J. Magana, A. (2020). Computational thinking in higher education: A review of the literature. *Computer Applications in Engineering Education*, *28*(5), 1174–1189. https://doi.org/10.1002/cae.22295

Poulakis, E., & Politis, P. (2021). Computational Thinking Assessment: Literature Review. *Research on E-Learning and ICT in Education*, 111–128. https://doi.org/10.1007/978-3-030-64363-8_7

Rom An-Gonz Alez, M., P Erez-Gonz Alez, J.-C., & Jim Enez-Fern Andez, C. (2016). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*. https://doi.org/10.1016/j.chb.2016.08.047

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Sondakh, D. E., Osman, K., & Zainudin, S. (2020). A Pilot Study of an Instrument to Assess Undergraduates' Computational thinking Proficiency. *International Journal of Advanced Computer Science and Applications*, *11*(11), 263–273. https://doi.org/10.14569/IJACSA.2020.0111134

Specht, M. (n.d.). *Professor Marcus Specht Keynote Speaker CTE 2020 | Centre for Education and Learning*. Retrieved March 17, 2022, from https://www.educationandlearning.nl/news/professor-marcus-specht-keynote-speaker-cte-2020

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers and Education*, *148*, 103798. https://doi.org/10.1016/j.compedu.2019.103798

Taslibeyaz, E., Kursen, E., & Karaman, S. (2020). How to Develop Computational Thinking: A Systematic Review of Empirical Studies. *Informatics in Education*, *19*(4), 701–719. https://doi.org/10.15388/INFEDU.2020.30

Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. *ACM International Conference Proceeding Series*, 120–129. https://doi.org/10.1145/2999541.2999542

Vinu Varghese, V. V., & Renumol, V. G. (2021). Assessment methods and interventions to develop computational thinking - A literature review. *2021 International Conference on Innovative Trends in Information Technology, ICITIIT 2021*. https://doi.org/10.1109/ICITIIT51526.2021.9399606

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). *Computational thinking*. *49*(3), 223. https://doi.org/10.1145/1999747.1999811

DOI: 10.34641/ctestem.2022.473

Developing a Continuous, Rather Than Binary, Classification for Measuring STEM Jobs

Ted CARMICHAEL[1*], John STAMPER[2], John CARNEY[3]
[1]TutorGen, Ft. Thomas, KY, USA
[2]Carnegie Mellon University, Pittsburgh, PA, USA
[3]MARi, Alexandria, VA, USA

ted@tutorgen.com*, jstamper@cs.cmu.edu, john.carney@mari.com

ABSTRACT

This paper presents our review and synthesis of the literature on STEM classification, and our results for a novel approach towards understanding, categorizing, and tracking STEM attributes in the workplace. We found two deficiencies in the way STEM is traditionally discussed, which we attempt to address in this work. The first is that the key components of STEM tend to be discussed holistically in the literature, rather than discretely as Science, Technology, Education, and Mathematics. The second is that our ability to track changes in S.T.E.M. concentrations in the workplace, both geographically and temporally, is underdeveloped. Further, we have found that this second deficiency is due, in part, to how STEM occupations are categorized; i.e., "STEM" tends to be a binary designation, rather than measured on a continuum for each job, and each component of S.T.E.M. It is also due to the lack of a "gold standard" measurement of the quantity of S.T.E.M. for all occupations. Here, we present a novel approach for machine learning algorithms using a "bag of words" method. These algorithms are trained on a small selection of Standard Occupational Classification (SOC) occupations, using ratings for each component of S.T.E.M. as the exemplars on which to train (SOC 2019). Recognizing that such a classification scheme is new, and that one of the goals of this project is to solicit Subject Matter Expert (SME) feedback, the resultant model of S.T.E.M. measurements across these occupations is designed to easily incorporate multiple distinct models and alternative approaches.

KEYWORDS

STEM classification, machine learning, Standard Occupational Classification (SOC)

1. INTRODUCTION

This report is one part of a greater research effort started through the U.S. Department of Education (DOE), Office of Career, Technical, and Adult Education (OCTAE) to create a STEM Index to quantify the elements of S.T.E.M. (i.e., science, technology, engineering, and math) in the North American Industry Classification System (NAICS), the Standard Occupational Classification system (SOCs), and possibly to the level of individual jobs. This statistical model will be designed to be applied to government employment and industry datasets to provide STEM educators and administrators information related to the amount of Science, Technology, Engineering, and Math within a range of occupations included in both SOCs and NAICS. One of the most valuable applications of this STEM index will be the future ability to match state and local programs and course completions to quarterly regional Bureau of Labor and Statistics (BLS) economic activity, thus allowing greater alignment between course offerings and regional in-demand jobs. (Note: our paper follows the standard nomenclature of using "STEM" when speaking about programs and initiatives holistically, and "S.T.E.M." when referring to the individual component.)

One key finding from our literature review is that the STEM categories are not generally broken down into composite classifications. Further, all occupations tend to be categorized as either STEM or not, based on broad definitions, and in a binary manner, rather than by quantifying the amount of STEM knowledge needed for a job or determining a list of required STEM skills and competencies. With the current classification method, some occupations will be included when they should not be, while others are excluded even if they have a significant STEM component. "Broad groupings can only give broad estimates and are not useful for targeted workforce policy. [...] Problematic in the current discourse on the value and impact of STEM discipline-related skills is the use of the STEM acronym to encompass a wide variety of different concepts in instances where a more precise or appropriate term is needed" (Siekmann & Korbel, 2016).

Finally, the current classification scheme itself resists timely updates. To assign classifications across such a comprehensive list of occupations is quite labor intensive. And there is a significant need to be able to compare data, both "across agencies and organizations," as well as over time, in order to "maximize the comparability of data" (SOC, 2019). This hampers our ability to make systemic changes in the classification and quantification of the STEM content across occupations.

Another pattern that we have found, not only in the STEM literature, but also found in many STEM-focused education websites, program initiatives, and advocacy resources, is a general trend towards defining STEM more broadly, rather than with more precision. That is, there is a marked tendency to discuss STEM skills holistically; and to promote STEM education, training, and directed resources by listing many that are more correctly classified as "foundational" skills. Foundational not just for STEM occupations, but for a wide variety of non-STEM

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

occupations as well. These are skills such as: creativity, organization, communication and teamwork, problem solving, and critical thinking. These skills remain far from unique to STEM occupations. Further, in some STEM resources, even more general traits and habits are discussed in this context, such as: persistence, flexible thinking, empathy, engagement, and metacognition (Costa, 2008; Asunda & Weitlauf, 2018).

From the Department of Education report: STEM Education Strategic Plan 2018: "Over the past 25 years, STEM education has been evolving from a convenient clustering of four overlapping disciplines toward a more cohesive knowledge base and skill set critical for the economy of the 21st century. The best STEM education provides an interdisciplinary approach to learning, where rigorous academic concepts are coupled with real-world applications and students use STEM in contexts that make connections between school, community, work, and the wider world. Leaders in STEM education continue to broaden and deepen its scope and further transcend the fields of study beyond just a combination of the four disciplines to include the arts and humanities. Modern STEM education imparts not only skills such as critical thinking, problem solving, higher order thinking, design, and inference, but also behavioral competencies such as perseverance, adaptability, cooperation, organization, and responsibility." This, as evidence that STEM is more and more often discussed as a synthesized field rather than as individual components; and more than just S.T.E. and M. skills are referred to as part of the STEM curricula.

This literature review and project takes as its working hypothesis the possibility that the pendulum has swung too far. Perhaps what is needed is a new focus on the individual pieces that comprise STEM; or, further, the four components of S.T.E. and M. (The convention is, when discussing STEM as a unified subject, the acronym is written as a single word; conversely, when emphasis is on the individual components, the acronym is written as "S.T.E.M.") This more granular focus may renew our understanding of all the skills, abilities, aptitudes, and competencies that go into performing STEM tasks and occupations. By more precisely measuring the competencies that comprise S.T.E.M., we might better track these skills across a continuum, and determine their changing proportions over time and across all occupations.

2. BACKGROUND

In recent years, educational and vocational professionals have sought to define STEM core competencies in a more holistic way. While emphasis on the sciences dates back at least to the decade that saw the formation of both NASA and the NSF, the acronym STEM, and the emphasis on these four fields - Science, Technology, Engineering, and Mathematics - emerged in the late 1990's and early 2000's (Chute, 2009). Coined by Dr. Judith Ramaley, who served as the assistant director of the Education and Human Resources Directorate at the NSF, "STEM" was defined as "an educational inquiry where learning was placed in context, where students solved real-world problems and created opportunities—the pursuit of innovation" (Daugherty, 2013).

What's more, there is a persistent belief that "only some kids can really learn math and science to high levels" (Chute, 2009). Here is Dr. Nancy Bunt, program director of the Math & Science collaborative in the Allegheny Intermediate Unit in Pittsburgh, PA: "There is this very strong belief out there on the part of parents and the part of some educators and society as a whole: If I wasn't good at math, my kids don't have a chance of being good at math. It's a gene thing. They'd never say that about reading. There is an assumption that everyone needs to learn to read." There is even data to support the idea that a strong emphasis on the four S.T.E.M. components, as well as their integration into the unified "STEM" category, with meaningful overlap and synergy, will not only increase the number of students who pursue the sciences, but also positively influence the number that pursue any bachelor's or post-secondary degree (Chute, 2009).

However, definitions of STEM in terms of what fields are included have tremendous consequences for US (and international) policy, funding decisions, resource allocations, and an incredible variety of educational initiatives and workforce development programs. Which fields of study are included can impact anything, from which among the millions of undergraduate and graduate students are supported by the NSF, to who will be eligible to receive student visas, to which programs receive extra resources as fields of designated national interest.

Consider, for example, the STEM Educational Act of 2014. Passed in July of that year, this act has only two stated purposes. Aside from a few minor changes in wording of the existing statues, this law was written solely "To define STEM education to include computer science, and to support existing STEM education programs at the National Science Foundation." And the second of these was not really a change; rather, it was just a continuation of previous policy: the bill states that "The Director of the National Science Foundation, through the Directorate for Education and Human Resources, shall continue [emphasis added] to award competitive, merit-reviewed grants to support" STEM learning environments, learning outcomes, engagement, and research in STEM education.

In other words, so crucial to funding decisions, programs, and resource allocation was explicitly adding "computer science" and computational thinking to the standard, government-wide definition of STEM that this act was passed by both houses of Congress and signed by the President, to make this inclusion clear and give it the force of law. Definitions matter.

As such, there is continual pressure, from stakeholders, curriculum managers, and workforce development programs, to include their individual fields and domains in standard definitions of "STEM," in order to remain relevant (and funded). Conversely, there is pressure in the other direction to keep STEM a semi-exclusive and useful

definition: becoming too broad would render it nearly meaningless. This reality has created tremendous ambiguity in the STEM label, such that it is often defined and redefined based on various needs, creating the situation where there is little agreement, across all stakeholders, in a precise definition.

The power of the STEM acronym comes, in part, from this ambiguity, as it can mean all things to all people. As a loosely defined and malleable concept, "STEM" gains wider acceptance and becomes the focus of more initiatives, more funding, and greater involvement of key stakeholders. However, this same ambiguity, that allows much of this national (and international) attention, also drives misunderstandings and confusion over what, exactly, STEM refers to. "Whether the acronym is understood and fashionable outside these education groups is not well known. What is known is that the acronym and associated term is not well-defined, even within groups that make heavy use of it" (Daugherty, 2013, citing Storksdieck, 2011).

But such ambiguity has consequences. The NSF's "STEM Education for the Future: A Visioning Report" from 2020 makes the point that access to STEM education varies "across zip codes and income levels," as well as among underrepresented groups (Education & Human Resources, 2020). It further articulates certain key priorities, in order to meet the challenges in STEM education and workforce development. One of these priorities is to level the playing field for STEM educational opportunities. And so, priority one is to increase opportunities for those being left behind.

Related to this, another listed priority emphasizes the need to motivate improvements across the board: to continue to strive for advances in our national capabilities; to promote increased invention and innovation; and to fill the demand for the high-tech, high-quality jobs of the rapidly approaching future. This is because, according to the National Science Board's Science and Engineering Indicators 2018, Americans' basic STEM skills have only modestly improved over the past two decades. And, they continue to lag behind many other countries. Further, according to the indicators, from 2006–2015, American 15-year-olds still tended to score below the international average in mathematics skills, and at or slightly above the international average in science skills. These are important areas to address.

However, having an insufficient definition of S.T.E. and M., and not going far enough in classifying STEM occupations vs. non-STEM occupations, means that, not only are we not accurately measuring these problems, but we lack the data to properly target interventions, and do not have the means to measure or judge the success or failure of those interventions.

What we need to do—the motivating premise behind this project—is to move away from the binary classification of STEM vs. non-STEM jobs, and instead focus on the level of STEM skills and abilities that are found within all jobs. This change in emphasis would improve measurement of STEM skills in our workforce in a way that is more granular, and thus provide a better understanding of which STEM skills and competencies are increasing in demand, so that we might better meet current and emerging workforce needs.

Our research in this project will look specifically at novel ways to do exactly this. The current definitions simply are not granular enough, and are not updated frequently enough, to allow more precisely targeted interventions. It is our hope that not only will we be able to measure and track a continuum of STEM-related skills needed for more precise categories of occupations, but also to quantify the changes in demand for these skills over time.

3. APPROACH

In order to develop a scalable, algorithmic method for quantifying each of the S.T.E.M. components across all occupations, two things are needed. First, we need a ranking of S.T.E.M. content for each occupation on which to train that algorithm. The ranking system should have both a theoretical and valid base. Second, we need a model to predict the rankings. Our approach to the model is that a single model is unlikely to provide a highly accurate result across all jobs, so we rely on a ensemble approach (Niculescu-Mizi et al., 2009) described in section3.2. and an initial first predictive model for the ensemble described in section 3.3.

3.1. *Rating System*

When considering how much Math, or how much Engineering, is required for a particular job, there are really two different senses for how to quantify this requirement: level of expertise needed to perform the job; and level of intensity, or how much time is spent on that activity. We chose to use "level of expertise" only in quantifying the amount of S.T.E. or M. required. So, for example, a retail job where one is expected to add and subtract figures all day would still have a low level of Math required, perhaps a 1 or a 2 on the 9-point rating scale.

Table 1. STEM Level Classification* Ratings.

Rating	Education Equivalent
0	None
1	Middle School
2	High School
3	Certification
4	Assoc. Degree (2 yr)
5	Bachelor's Degree (4 yr)
6	Master's Degree
7	Doctoral Degree
8	Everything Above

*These designate minimum levels (or equivalent) of knowledge acquisition via degrees and/or years of experience.

Conversely, a job that requires Calculus and Differential Equations knowledge would be rated high, even if the employee was not expected to draw on those skills very often. In this way our system quantifies the level of skill

needed to perform the job, so that qualifications and how they change over time are captured.

The STEM level classification was done on a 9-point scale, with the level of rating roughly equivalent to years of education as a proxy for knowledge requirement to perform the job. Table 1 is included to illustrate these rating levels.

3.2. *Ensemble Approach*

In order to achieve robust results, we have implemented a workflow that includes an ensemble methodology (Niculescu-Mizi et al., 2009, Yu et al., 2010). This methodology recognizes that any single model might not be highly accurate across all possible predictions, and by combining models using weighted averages a better result can be achieved. Our ensemble scoring methodology can be seen in figure 1, and is available in source code made available in our public repository https://github.com/jcstamper/CTE-STEM.

$$\text{IndexScore}_{s,t,e,m}(\text{SOC}) = M_1\omega_1 + M_2\omega_2 + \ldots + M_x\omega_x$$

For Models [M] and Weights [ω]

where $\sum \omega = 1$

Figure 1. Ensemble workflow for weighting models.

3.3. *Model Prime*

We proposed and implemented an initial model for our workflow that we named M_{Prime}. This model uses NLP methods in a bag of words approach from a data source that contains job descriptions and compares word embeddings against a weighted vector for each of the formal topics of Science, Technology, Engineering, and Math. The vector distance is then calculated and normalized to our ranking scale.

For our first pass, we derived the vectors for the formal topics from a list of topics curated by STEM experts, and our initial data source for the job descriptions came from O*NET.

4. RESULTS

We ran our initial model on 82 jobs exported from the O*NET repository. Note that because we do not have a true gold standard for our topic vectors, our weightings were not trained in any way. Having more data from additional models or from experts could help us better train the models in the future. The results, however, were promising. Although experts were able to find potential disagreements in the results, we compared Cohen's Kappa statistic on 25 jobs classified by two experts and our system. The results between the two experts was .55 and between the two experts and our system were .52 and .44 respectively. While all these values suggest a weak level of agreement, it also shows that our values were not far off.

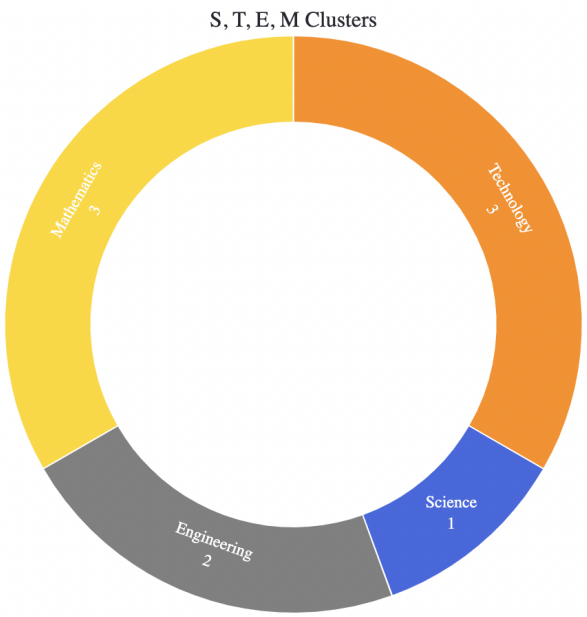


Figure 2. Index for SOC 27-1022, Fashion Designers, as a pie chart visualization.

We created an interface and web portal to inspect our results, which are available here in our web application https://share.streamlit.io/jcstamper/cte-stem/main/AppFinal.py. An example of selected jobs with several visualizations can be seen in Figures 2 and 3.

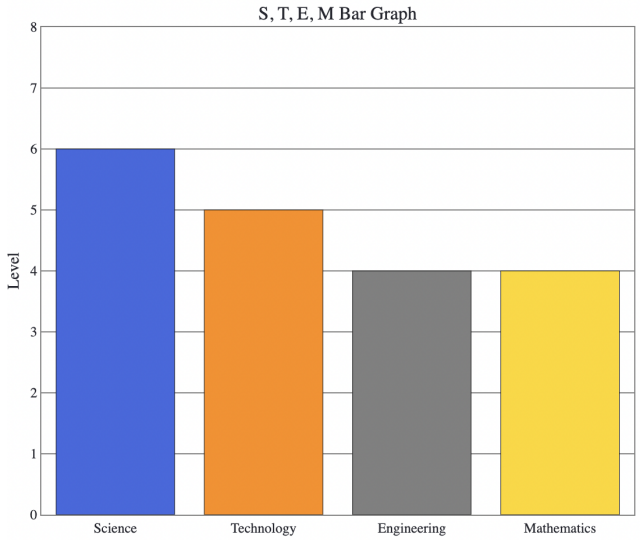


Figure 3. Index for SOC 11-101, Chief Sustainability Officers, as a bar chart visualization .

5. DISCUSSION

There are two primary stakeholders groups for robust S.T.E. and M. classifications: employers with non-STEM jobs, and educational institutions preparing students for the workforce. Currently, middle and high school STEM activities, curriculum, and specialized STEM academies focus on STEM jobs and occupations. STEM funding and a significant amount of school resources are directed towards STEM. Non-STEM jobs and occupations, with no S.T.E.M. educational requirements, potentially have key

gaps in the curriculum that results in proficiency gaps for entry-level employees.

With a framework established to facilitate a gold standard for an S.T.E.M. rating system, we believe that industry-specific subject matter experts will be able to quickly fine tune the algorithms to provide road maps for job candidates and educational institutions.

Future applications of the S.T.E.M. index could potentially allow K-12 school districts to automatically match their non-STEM career and technical education offerings to the latest Bureau of Labors Statistics economic data for their region to gain insights into how aligned their S.T.E.M. curriculums are to the economic needs of their region.

6. CONCLUSIONS

The U.S. Department of Education (DOE), Office of Career, Technical, and Adult Education (OCTAE), and other stakeholders, have identified a need to create a STEM Index, quantifying the elements of S.T.E.M. (i.e., science, technology, engineering, and math) in the workforce. By more precisely tracking required S.T.E.M. skills as they change over time, and as differences appear across geographic regions, we will be better positioned to respond to education and training needs.

This is preliminary work; a first step in generating a new STEM Index. Our goal is to create a starting point for SMEs and stakeholders to contribute in meaningful ways, and as such our approach for integrating alternative methods and models is a key outcome of this project. We hope people will inspect and scrutinize the algorithms, ratings, approaches, and outcomes we have developed, which are here: github.com/jcstamper/CTE-STEM. And we also want to see other innovative approaches, and gather additional stakeholder requirements and use-case examples. In particular it is important to find novel ways to utilize the vast amounts of data currently being collected, as well as identify new data sources that need to be developed.

7. ACKNOWLEDGEMENTS

This document was produced and funded at least in part with Federal funds from the U.S. Department of Education under U.S. Government Contract Number: 919900-21-C-0011. The content of this report does not necessarily reflect the views or policies of the U.S. Department of Education nor does the mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government. Albert Palacios served as the contracting officer's technical representative.

8. REFERENCES

Anderson, D. M., Baird, M. D., & Bozick, R. (2018). Who Gets Counted as Part of America's STEM Workforce? RAND Corporation8.

Asunda, P. A., & Weitlauf, J. (2018). STEM habits of mind: enhancing a PBL design challenge-integrated STEM instruction approach. Technology and Engineering Teacher, 78(3), 34-38.

Chute, E. (2009). STEM education is branching out: Focus shifts from making science, math accessible to more than just brightest. Pittsburgh Post-Gazette, 10.

Costa, A. L., & Kallick, B. (2008). Habits of mind in the curriculum. Learning and leading with habits of mind, 16, 42-58.

Daugherty, M. K. (2013). The Prospect of an "A" in STEM Education. Journal of STEM Education: Innovations and Research, 14(2).

Donsbach, J., Tsacoumis, S., Sager, C., & Updegraff, J. (2003). O* NET analyst occupational abilities ratings: Procedures. DFR-03-22). Alexandria, VA: Human Resources Research Organization..

Education & Human Resources (2020). STEM education for the future: A visioning report. National Science Foundation.

Fleisher, M. S., & Tsacoumis, S. (2012). O* NET analyst occupational abilities ratings: Procedures update. Alexandria, VA: Human Resources Research Organization.

Fleisher, M. S., & Tsacoumis, S. (2012). 'O* NET analyst occupational skills ratings: Procedures update. Raleigh, NC: National Center for O* NET Development.

Johnson, C. C., Peters-Burton, E. E., & Moore, T. J. (Eds.). (2021). STEM Road Map 2.0: A Framework for Integrated STEM Education in the Innovation Age. Routledge.

National Center for O*NET Development. O*NET OnLine. Retrieved July 26, 2021, from https://www.onetonline.org/8.

Niculescu-Mizil, A., Perlich, c., Swirsz, G., Sindhwani, V., Liu, Y., Melville, P., Wang, D., Xiao, J., Hu, J., Moninder, S. (2009). Winning the KDD Cup Orange Challenge with Ensemble Selection - Journal of Machine Learning Research, W&CP 7, pp. 23-34.

Oleson, A., Hora, M., & Benbow, R. J. (2014). What is a STEM job? How different interpretations of the acronym result in disparate labor market projections. Viewpoint paper, 1.

Rothwell, J. (2013). The hidden STEM economy. Metropolitan Policy Program at Brookings.

Siekmann, G., & Korbel, P. (2016). Defining "STEM" skills: review and synthesis of the literature. Support document, 2.

SOC: Standard Occupational Classification Manual (2019). Attachment A: Options for Defining STEM occupations under the 2018 SOC system; SOC Policy Committee recommendation to the Office of Management and Budget.

STEM Education Act of 2014. (2014, July 15). Text - H.R.5031 - 113th Congress (2013-2014): https://www.congress.gov/bill/113th-congress/house-bill/5031/text.

Storksdieck, M. (2011). STEM or STEAM. The Art of Science Learning. Retrieved October 14, 2011, from http://scienceblogs.com/art_of_science_learning/2011/04/stem_or_steam.php.

Costa, A. L., & Kallick, B. (2008). Habits of mind in the curriculum. Learning and leading with habits of mind, 16, 42-58.

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35.

Tsacoumis, S., & Willison, S. (2010). O* NET® Analyst Occupational Skill Ratings: Procedures. Alexandria, VA: Human Resources Research Organization.

Yu, H¬F., Lo, H¬Y., Hsieh, H¬P., Lou, J¬K., Mckenzie, T.G., Chou, J¬W., et al., (2010). Feature Engineering and Classifier Ensemble. Proc. of the KDD Cup 2010 Workshop, 1-16.

DOI: 10.34641/ctestem.2022.474

Computational Thinking, History and Non-formal Learning-A Well-crafted Blend!

Irene SILVEIRA ALMEIDA*, Ajita DESHMUKH

Goa University, India,
MIT-ADT University, Pune India
irene@unigoa.ac.in, ajitadeshmukh13@gmail.com

ABSTRACT

Computational thinking (CT) is one of the core skills required for 21st century education. As we transition from STEM to STEAM by incorporating Art, it becomes imperative to see the application of CT in Humanities. The subskills of CT can be integrated in both -formal and informal teaching-learning practices of Humanities. This paper studies guided learning practices by the instructors that enable application of CT subskills of decomposition, pattern recognition, and abstraction by students of Post Graduate History Course. This qualitative study explores development of CT skills through Historical thinking using a specifically created WhatsApp group as a communication channel for the purpose of fostering and guiding Computational Thinking Skills. This study further explores how informal learning through WhatsApp communication aids in the development of CT skills of abstraction, pattern recognition in the process of discussions and Historical thinking. The qualitative analysis of WhatsApp posts illustrate how the CT skills are nurtured and applied by students without any formal knowledge of the same. Higher frequency of learner-to-learner messaging mirrors watercooler and corridor communication and dramatically moves learning away from solely instructor-learner directional communication chains specific to traditional learning spaces. This corroborates the importance of free and informal learning space in the development of CT skills in students beyond the domain boundaries of STEM.

KEYWORDS

Computational Thinking, CT in Humanities, non-formal learning, WhatsApp groups, SMS language.

1. INTRODUCTION

In today's technological world, computational thinking comes across as a concept of increasing importance. Given the need in contemporary societies for citizens who can understand, control and work with technology, this field acquires great relevance. Computational thinking has been making inroads into education and is being advocated at diverse schooling levels. Although initially linked to the domain of computer sciences, computational thinking has deftly crossed over to other disciplines. Jeannette Wing (2006) contributed to the popularization of computational thinking and viewed it as a fundamental skill on par with reading, writing and arithmetic. Computational thinking involves thought processes that are used to solve complex problems. The problem-solving skill is considered a universal skill in the 21st century and has supplanted the focus on rote skills. In this context, it is recommended that students be exposed to it as they engage with core disciplines in their educational trajectory. However, computational thinking and its sub skills have been assuredly less prominent in humanities than in traditional STEM domains. Computational thinking has rich cross-curricular potential and its appropriation in STEAM domains attains particular relevance in this respect (Merino-Armero, J.M, 2022). Recent research in computational thinking integration into language classes has shown that narrative content demanding interpretation lends itself well to computational thinking activity (de Paula, 2017). Researchers and practitioners are increasingly working towards integrating computational thinking into curriculum in a broad spectrum of disciplines spanning graphic art, languages, humanities, astronomy, history, geology, biology etc. (Settle, 2012, Czerkawski, 2015). Studying History involves familiarizing and dealing with large amounts of information. Learning of History is based on Historical Thinking that typically involves multiple CT skills. Unfortunately, students in a History class often remain passive listeners and consumers of content related to historical figures, events and chronology leading to rote memorization. Introducing content necessitating computational thinking into History is challenging and calls for innovative pedagogical practices such as integrating informal learning into formal learning. Gunbatar (2019) lists decomposition, abstraction, algorithm design, debugging, iteration, and generalization as a set of subskills. Decomposition involves separating the problem into manageable steps. Computational thinking also calls for representing data through abstraction such as models and simulations, all of which are applicable to learning of History. Integrating computational thinking in History class necessitated modifying this approach to an active engagement with content. Widespread availability of digital texts makes it easy to obtain information. However, students often lack skills in interpreting the texts and arriving at valid conclusions. Puzzles and enigmas were used to provide a problem-solving approach to an otherwise passive acquaintance with historical events. Puzzle-solving requires information gathering, sifting through data, making intuitions, testing intuitions and making sustainable claims as a solution. Decomposition, abstraction and pattern recognition constitute vital strategies for efficient puzzle-solving. The development of CT skills as integrated in learning of History is of critical importance today since these will also effectively help students to identify fake news, critically analyze the stories circulated over social media and otherwise enable them to pick fact over fiction. These skills are the most sought-after skills in various industries and

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

jobs of the technology-driven society, thereby preparing them to be 'industry-ready'.

2. METHODS AND PRACTICE

The present research is an attempt to blend historical narratives with computational thinking sub skills in an informal e-learning environment. Given that computational thinking is inherently a problem-solving approach, content generally taught in History class was reworked into historical narratives with inbuilt puzzles and enigmas. The implementation of the study began in December 2017 as part of the 4-month semester spent studying Representation of French History in Visual Art and Literature in Goa University, India. 11 students enrolled in the course and were part of the study. The analysis of this study involved qualitative analysis of the WhatsApp group Chat. The chat was analyzed for sentiment and content after the course. The participants were anonymized for the study by giving them codes such as HS1, HS2, HS3 and so on. This study does not study the development and/or application of CT skills as well as communication within the WhatsApp group with respect to gender. The WhatsApp group-activity that forms the basis of the project was complementary to classroom discussions and spanned the entire duration of the course. In a pre-pandemic era where teaching was primarily in offline mode and most teaching activity restricted to the four walls of the classroom; this mode of communication was chosen with the primary objective of channelising the sustained collaborative engagement of familiar chat room scenarios towards the cultivation of problem-solving skills. The familiarity and informality, any-time, seamless and cross member communication would offer a platform to boost autonomous fact finding, knowledge acquisition and problem solving leading to collaborative learning among the group. The resulting group cohesion could assist slower learners in developing computational thinking sub-skills like decomposition, abstraction and pattern recognition.

3. RESEARCH QUESTION

This explorative qualitative study is guided by the following research questions: Does Historical thinking lead to development of CT skills and vice versa? Does informal learning, especially using social media aid the development of Historical thinking and thereby CT skills? How does Historical thinking develop CT skills, especially of Pattern Recognition and Abstraction?

4. THE STUDY

A WhatsApp group of 11 students and the instructor was specifically created. The student members were given orientation of the functioning of the group prior to the enrolment in the group and were encouraged to freely use the space for class related communication. In addition to functioning as a notice board (announcements related to class venues, timings, test schedules and modalities, exam preparation etc.), the WhatsApp group was designed to function as an informal space where content related to history would be decomposed by students in a fun and competitive set-up.

To begin with, students were informally provided questions on History. Correct answers were rewarded with a thumbs up emoticon (👍). Informal positive reinforcement was then used to motivate learners to solve history-based puzzles and enigmas. At times, clues were provided and learners were gently prodded into pattern recognition by keywords and vital links in information. Students attaining 20 thumbs- ups were declared winners, leading to applause and appreciation from all. The informal ambiance was carried over in class as well: students were encouraged to interrupt, ask questions, and discuss during classes. The WhatsApp group acted as a supplement to this informal styled learning of French history and allowed for extended learning hours and freedom to engage with a wide array of sources as per individual choices. It is pertinent to note that the puzzles and enigmas were generally posted on WhatsApp prior to class discussion on the topic. Preference for a flipped classroom approach ensured higher levels of engagement and authentic problem-solving versus a passive revision/ recall. Carefully crafted questions announced puzzles and enigmas through an original format of common-place narratives. Since the narratives were laden with explicit and implicit meaning, they would often demand interpretation. Participants were thus nudged towards computational thinking as they opted to eliminate irrelevant details, search for defining patterns and work backwards towards the solution.

5. PUZZLE DESIGN

Initial questions were kept fairly straightforward and simple so as to create comfort levels and build confidence. Gradual introduction of elaborate puzzles and enigmas led to deeper reflection and use of computational thinking sub skills in the quest to type out the first correct-response post. Through trial-and-error, observation and practice, the complex puzzle-sets drove acquisition of problem-solving skills and ensured that students comprehended, interpreted and re-verbalized information to fulfil puzzle-solving norms. Earlier response styles replicated problem solving skills at a fairly basic level: students resorted to identifying key terms, searching for related information, finding the answers and posting the content verbatim. Such learning patterns were slowly discouraged. Students were steadily weaned away from copy-pasting content and prodded into drawing their own conclusions by such humorous posts: "*But this is the last time Wikipedia scores… Next time we want you answering/ It wasn't Wikipedia this time*😑*/ Miss Wikipedia not a member of this group"*. Instructors emphasized on development of complex problem-solving skills and crafted question posts accordingly. Simplistic answers were disqualified and interpretation was necessitated, valued and incentivised. The basic guiding principle being computational thinking skills through historical narratives, the question posts were appropriately designed to lead participants toward targeted subskills like pattern recognition, decomposition and abstraction. Reproduced below are a few examples of question posts.

6. DISCUSSION OF WHATSAPP CHATS

The WhatsApp posts reproduced below are verbatim. No corrections in spelling or grammar have been made. Emoticons have been retained.

6.1. Pattern Recognition

In one of the initial puzzle-sets, major historical events from the 12th to 14th centuries were dealt through decomposition. Question posts introduced dramatic events surrounding key historical figures of the time. Framing of questions around a central thematic thread ensured that responses could be obtained through pattern recognition. Acquaintance with historical facts, sifting through extensive information, and deciphering the narratives were key steps in puzzle-solving. In the example below, the pattern is constructed around KING and later extends to QUEEN and /PRINCE. Posterior puzzles would build on QUEEN and PRINCE; supplementary unnumbered question posts on QUEEN and PRINCE conclude the set and function as a transition toward the subsequent Puzzle set.

"1. Which king is a Saint? 2. And which King's envoy slapped the Pope? 3. Which king owed his crown to a teenage girl? 4. Which king dealt with his enemy by making him son-in-law? 5. Which king divorced his queen because she bore only daughters?AND what happened subsequently to the ex-queen?And... who is the black/dark prince?"

As seen from the student reply posts below, multiple students have attempted to answer in random order. Considering this discussion, they were required to look up on the Internet (out-of-class), and correctly match historical figures to each of the patterns. Patterns were built around motifs of major players on the French historical scene in the medieval period- King, Queen, Pope, Prince, Saint, Commoner/peasant girl. The motifs were augmented with action–based relations between the players: King becoming Saint, King defying Pope, Commoner girl assisting King, King entering into marriage alliance with enemy King, King divorcing Queen in the absence of a male heir, Actions of the defiant Queen, Actions of a Dark enemy Prince. Recognising the pattern constituted the first part of the challenge, searching for details that correspond to the patterns and ignoring irrelevant details made up the second part. This challenge was based on pattern-recognition and understanding, and did not call for memorizing and recall. Content required for puzzle solving had not been dealt with earlier, but was easily reachable on the Internet through skilful use of keywords in search engines. Attentive reading and drawing links between patterns were required to obtain correct answers. Answers were obtained not by chance but through careful thought and understanding. Student posts provide precise reasons for the association of patterns with specific historical players, thus proving that they have successfully confronted both challenges.

"Edward IV also known as Edward of Woodstock is the black prince" (HS1).

"Charles VI makes Henry V his son-in-law" (HS3)

"Louis IX of France" (HS9)

"1- his compassion for the poor and suffering people and other acts of charity.

2.Known as saint Louis"(HS9)

"The Black Prince's Ruby was given to Edward III, by Peter of Castile, as an appreciation for his services in regaining the throne.

It is one of the oldest parts of the Crown Jewels"(HS5)

Only 3 of the 7 puzzles (42%) were correctly attempted, the remaining were ultimately explained in following class sessions. To improve on the students' ability to recognise patterns, a subsequent puzzle highlighted similarities and differences between protagonists. The pattern in the puzzle below is centred on WOMAN with its different categories- LADY, QUEEN, PRINCESS, NOBLEWOMAN, PEOPLE'S WARRIOR, MOTHER, WIFE… The puzzle draws on similarities (Qt 4- similar to Charles Martel, similar achievements/ Qts 1 & 3 relation to multiple (2-3) French Kings/ Qts 2&5- unexpected people attaining the crown, throne). The puzzle equally capitalizes on differences to induce pattern-recognition. (Queen to 2 French Kings v/s mother to 3 French kings, princess marrying a king and becoming queen v/s a mother marrying her son to a princess and making him king, Charles Martel associated with a hammer v/s Jeanne Hachette associated with an axe due to martial exploits). Similarities and differences are a combined force that drive the narratives and are conducive to the interpretation that is key to the WhatsApp puzzle-solving. Pattern-recognition thus yields result in problem resolution and not only later helps in recall during exam preparation but more importantly, finds application in Humanities Education.

"1. Which French lady was queen to 2 French Kings? 2. Which French Princess later became Queen of.... France? 3. Which Italian noblewoman became mother to 3 French kings? 4. Which Frenchwoman earned a nickname similar to Charles Martel for... similar achievements? 5. Which French noblewoman managed to put her son on the French throne?"

This puzzle-set registered a higher success rate than the previous one (50%), and an additional question (below in bold) was instantly thrown in to intensify the challenge. The chat that ensued shows that 2 puzzles were promptly answered, and another 2 led to sustained attempts until the correct answer was finally obtained. The attempts showcase claims based on pattern recognition and historical reasoning as students try to substantiate their claims. Debunking incorrect claims and arriving at the correct answer involved a further engagement with similarities and differences, thus reinforcing pattern recognition skills. (Mary is associated with Scotland, not France; Joana of Flanders is a noblewoman, although fiery, not connected to a weapon similar to Charles' hammar; in contrast to Mary of Scotland, Margaret of Valois is a French born Princess who marries a man who ultimately becomes King of France)

"3.Catherine de' Medici" (HS3)

"1. Eleanor of Aquitaine" (HS7)

"2. Marie Antoinette" (HS6)

"3. Catherine de Medici"(HS3)

"1. Anne of Brittany" (HS7)

"5. Judith of Brittany"(HS3)

"5. Judith of Bavaria" (HS7)

"2. Claude of France" (HS10)

"4. Not sure but is it Joan of Arc? She was nicknamed the maid of Orleans

For her role during the Lancastrian phase of the hundred years war" (HS8)

"No. Similar to Charles MARTEL"

"👍 Catherine de Medici 1. 👍 Anne of Britanny 2.👍 Claude de France".

"But there is one more answer. Anyone? ???"

"Judith incorrect. Keep trying for 4. & 5."

"2. Emma of France" (HS3)

"Now I feel it's Marie Thérèse of France".(HS3)

"No".

"2. Adelaide of Aquitaine" (HS3)

"No"

"Mary, Queen of Scots? She married the dauphin of France-Francis II. Later, In 1559, he ascended the throne as king so she would be queen Ans 2" 👆(HS7)

"But was she a french princess to begin with?"

"I don't think so. So wouldn't qualify"(HS3)

"She was Scottish. But wouldn't marrying the Dauphin make her princess?" (HS3)

"4. There is a Jeanne Hachette but she is a peasant's daughter."(HS6)

"While Joanna of Flanders is a noblewoman nicknamed Fiery Joanna"(HS7)

"Oh the name is Joan the Hatchet"(HS6)

"👍Jeanne hachette. The hatchet. 4".

"French born princess in french royal house. Then becomes queen of France. Strange coincidence"

"Margaret of Valois 2nd question" (HS10)

"Yes. Marguerite De Valois". 👏

6.2. Abstraction

The puzzle-set on WOMAN was followed by a single question post: *"Find the men behind these women. And mention the relation".* To arrive at answers to such a question, participants would be required to engage in extensive reading on the times and achievements of the 5 female protagonists. Computational thinking skills of abstraction would be predominantly used at this point. Through sustained practice, participants grew adept at focusing on the required information and ignored the many irrelevant details that habitually clog information gathering processes. Filtering of key details to obtain answers within short time spans meant that participants gained abstraction skills as they repeatedly engaged with such questions. Reproduced below are participant posts that bring out noteworthy trends in collaborative learning through abstraction. Instructor posts provide feedback that extends beyond mere categorizing of the response as being correct or incorrect. Instructor connects with earlier seen historical figures, encouraging pattern recognition or provides key information that guides abstraction. Instructor also creates a time bound challenge and pushes participants to simultaneously use multiple computational thinking skills in a race towards the correct response. The series of posts below feature multiple participants posting at very short time intervals, thus contributing to a near synchronous online learning experience in which individual thinking processes are visible and allow for systematic gains through observation, imitation and inference. Fragmented participant posts are indicative of real time application of computational thinking as students speed up response time.

The posts below highlight building and testing of abstractions as students propose claims with reasons, test them out until problem resolution. Most of History includes abstractions of histories and biographies as well as abstract projections of events of the real world. Abstraction evidently comes into play in the process of filtering out the details that are unimportant and hamper resolution (Charles IX, Louis III). In the case of Margaret of Valois, the initial option proposed, Charles IX, plays a secondary role in her elevation by arranging for her marriage. Her place in History is assured by her marriage to a man who would eventually become the first King of the House of Bourbon. Students would be required to compare and weigh both pieces of information before arriving at the conclusion that her husband Henry IV whose achievements are well-documented should be given precedence over her brother Charles IX, a weak and unpopular king. In the case of Catherine of Medici, it would be convenient to think of her third son as the source of her power, yet it is her husband, Henry II who through his premature death created a power void leading to her elevation and giving her a much-desired opportunity to do away with her rival, Diane of Poitiers. Her sons, Francis II and Charles IX reigned during the initial years following her husband's death and their youth/dependency provided her a golden opportunity to manipulate and exercise true power. In both cases, the more obvious answers are often discarded in favour of more justifiable stronger options. Exercising such choices forms an integral part of abstraction. Students were able to eventually choose the correct options over the more obvious information. A post that may go unnoticed is the discreet answer pertaining to Claude of France. The student appears to have successfully sifted out the irrelevant information (the fact that she was born to a French King, Louis XII, and was thus a princess of France) before providing the key point in the one-liner post: *For Claude of France - Francis I who made her his queen.* The wording of the answer displays confidence on behalf of the student who has successfully done the underlying abstraction needed to arrive at Francis I, the well-known Renaissance King who married Claude of France, thus changing her status from Princess of France to Queen of France. The student would have had to understand, through related readings, that this is far from the fate of most Princesses who go on at best, in rare cases, to become Queens (by marriage) to foreign Kings but hardly ever become Queens of their own country. It is commendable that the student utilized abstraction at multiple levels in a background

exercise before posting the correct answer in a single attempt.

Students also drew on previously acquired knowledge and skills to make connections, showcasing an interest for in-depth learning. Pattern recognition is intuitively being reutilised by a student who connects a historical figure from this puzzle-set to the previous puzzle-set (post in bold). The student introduces Mary of Scots, another well-known historical figure, in relation to Catherine of Medicis. "*Which Italian noblewoman became mother to 3 French kings?"* is a question from the previous puzzle-set that leads to the discovery of Mary of Scots, the wife of a son of Catherine of Medicis, the Italian noblewoman. *" Which French Princess later became Queen of.... France?"* from the preceding puzzle-set also serves as a lead to Margaret of Valois and Claude of France, two Princesses of France who through strange coincidences become Queens to Kings of France. This implicit understanding would be vital in the abstraction that becomes necessary to arrive at the correct answers involving the men behind these female protagonists in the current puzzle set.

"Can u consider Charles IX the Impt man behind margret of Valois.? HE was one of her brothers who arranged for her to marry king Henry III of Navarre who would later go in to become first Bourbon king of France .

So technically he was responsible for her position" (HS4)

"He is not of considerable importance. And not the one who made her queen"

"She owes her queenship to which man?"

"Umm...ber husband?"(HS4)

"Name. Hurry up"

"Henry III of Navarre Later Henry IV Of France"(HS4)

👍

"for Jeanne hatchette, is jt Louis IX ? Cz he acknowledged her heroic deeds by instituting a procession called "procession of the assault". also married her to her lover". (HS2)

"Is it IX?"

"XI Sorry". 😔(HS2)

👍

"For Catherine de Medici is it her third son Louis III?"(HS8)

"Incorrect"

"Catherine's husband Henry II" (HS11)

"I thought it's him but he would exclude her from evrythng nd shower favours on Diane de Poitiers" (HS5)

"For Claude of France - Francis I who made her his queen"(HS1)

"is Mary Queen of Scots the daughter in law of Catherine de Medici? I might be wrong".(HS5)

👍 *"Claude. Francis"*

"Henry finally died on 10 July 1559. She now enjoyed more power than she had as the queen and quickly started exercising her authority". (HS11)

"So it could b she benefited from his death"(HS11).

"Right. She did. So she exercised great power during whose reign? Thanks to which man? Men"

"Her sons Francis II and Charles IX"(HS11).

"Because they were very young when they became the king.(HS11).

She ruled as queen mother on their behalf".(HS11).

"And even though her son Henry III was an adult, he was dependent on her for politics and administration".(HS11).

"Correct. 3 sons. Kings. Francis II Charles IX Henri III"👍

6.3. Decomposition

Key events in the French Wars of Religion were dealt with in informal tones reminiscent of contemporary tabloid headlights. Solving the puzzle-set below would necessarily entail engagement with multiple sources and application of decomposition and historical reasoning.

"The bloodiest wedding in French History. 1. When? Date 2. Where? Exact site-venue 3. Bride and groom? 4. Why bloody? 5. Not a love marriage. Also UNLIKELY match. Why? 6. The end. Did they live happily ever after? 7. Bride's ex? 8. Did the ex or hubby pose a threat to the bride's Bro later? 9. Bride's relations with bros? Good or not? Explain 10. Mother of the bride? What party? 11. Party of bride's ex? 12. Party of groom's friends?"

Through this original puzzle-set, the entire topic (events of French Wars of Religion) was required to be decomposed into elements that fit the narrative. Grasping each of the decomposed bits led to the overall puzzle-resolution and an understanding of the entire topic. Participants successfully navigated the diverse events spanning decades to solve the puzzle which registered a 100 % success rate. In so doing, they independently figured out connections between events, reasons for and consequences to incidents, and identified the major players. The quest for such responses required supplemental related readings that stimulated an understanding of events that flowed decades later and which were not explicitly mentioned in the question posts. Interpretation of historical facts to fit in the requirements of the narratives entailed higher levels of historical thinking and problem-solving.

"1. 18 August 1572" (HS10).

"2. Notre Dame Cathedral in Paris"(HS1).

"3. Margaret of Valois and Henry of Navarre"(HS3).

"4. It was hoped this union would reunite family ties (the Bourbons were part of the French Royal family and the closest relatives to the reigning Valois branch) and create harmony between Catholics and the Protestant Huguenots..

Sorry that 5".(HS1)

"6. No. Their marriage was nullified".(HS9).

"7. Henry, Duke of Guise"(HS3).

"4. Bloody because of the clash between the catholics and protestants. It was also known as the French war of religion"(HS10).

"4. Bloody because six days after their marriage, the massacre of Protestants began on St. Bartholomew's Day"(HS11).

"6. Six days after the wedding - massacre of Huegeunots. Though Henry's life was spared, they were kept as prisoners, Henry

finally escaped n returned back to France but Marguerite was not allowed to follow him. Then in 1582 when she did go to bear an heir, but didn't succeed. Eventually goes to the Duke of Guise n tries to oppose Navarre's succession to the throne - no happy even afte4. After he became king their marriage was annulled n he married someone else"(HS3).

"10. Catherine de Medici. She was a Catholic".(HS7).

"9. Bride's relation with Henry III wasn't good"(HS8).

"11. Catholics"(HS3).

"12. Protestant Huguenots"(HS3).

"8. With no Valois heirs, Henry of Navarre was next in line to the French throne".(HS9).

👍 *"9&10"*👍 *"1, 3, 6, 7, 11, 12"*

👍 *"4. The Massacre a few days later. Hence bloody".*

"Qts 2, 5, 8 need more explanation and precision. Still open".

"2. They were wed outside Notre Dame cathedral because Henry was not Catholic".(HS3).

"Because she was in love with Henry I, duke of guise"(HS8).

"Also unlikely match because they were of different religions?"(HS3).

"Yes different and warring religious faiths at the time.. 👍 2. 👍"

7. OBSERVATIONS

7.1. CT Skills and Historical Thinking

Historical thinking and Problem solving involve lateral thinking over linear thinking. It also involves critical analysis and fact checking before proceeding. The absence of traditional class timings and the possibility of round-the-clock chatroom activity allowed for sustained efforts towards problem-solving. Students gained from interaction in terms of multiple perspectives and built upon these to consolidate learnings, to conclude and to arrive at the correct answers, in case of quiz or a puzzle.

It is observed that students do not rely on memorization or merely the prescribed textbooks but refer to multiple sources to draw a conclusion or respond which is the basis of Historical Thinking and is related to CT skill of decomposition. Only upon decomposition based on multiple historical resources can there be analysis and pattern-recognition. This also suggests the movement of students from being passive consumers towards actively engaged in the learning process of History. The students are actively involved in identifying similarities, trends, patterns and sequences in Historical events.

A steady progress was seen in the students' ability at puzzle-solving. Over time, decomposition and pattern recognition skills were developed, leading to prompt correct responses, identification of gaps in historical knowledge as well as critical analysis of the facts presented in the puzzle. It was observed that students mapped the historical event in contemporary events and attempted to relate the underlying socio-political thought processes and the evolution of these thoughts and society as a whole. This also demonstrates the skill of abstraction and deep analysis of trends of society where the students kind of create a mental timeline- an algorithm of sorts- if not as a timeline as an artefact. This demonstrates the application of CT skills in the Humanities domain.

A distinctive improvement in the way students handle data is observed.

7.2. Classroom Benefits of applying CT Skills

Inclusion: Students who were initially slow learners in class and showed disinterest in the subject were drawn by the WhatsApp group's informality and quizzing and manifested a strong presence in the chatroom. By observation, imitation, trial-and-error, they learnt to use decomposition and pattern recognition in the quest for correct answers.

High Motivational levels and Student Engagement: It is established that it is very tricky to calculate 'student engagement', especially in the context of social media used for educational purposes. Few of the crucial parameters are: the time of engagement, duration of engagement and nature of engagement. Engagement with the subject matter/ content taught outside the classroom hours, especially when it is not related to completion of assignments/ homework is an indication of higher levels of engagement. The WhatsApp group witnessed high student activity outside classroom timings and well into night time. Competition and time constraints played an important role in promoting problem-solving. Following a question, the delay between posts was often short and sometimes less than a minute. Running a race against time often prompted students to post "*Wait/ trying/ finally/No wait/"* indicating their involvement, engagement and the competitive edge. It is also observed from the above instances of expressions that students were comfortable in expressing their emotions which is a parameter of engagement (Appleton.J., Chrisenson.S., Kim.D., Reschley.A., 2006).

8. CONCLUSION

Teaching- Learning History with CT has been helpful in multiple ways as seen in this study. Firstly, the fast-paced posting and the direct inter-student communication flow was indicative of the success of collaborative learning. Secondly, it provides for the application of disciplinary concepts, analysing evidence and evaluating sources. Computational thinking serves as a toolkit for development of historical thinking. Integrating CT skills empowers History students and also, largely social studies students and curriculum to have a far-reaching and flexible approach and speak the language of the 21st century.Additionally, response precision and speed further corroborated the efficacy of group based learning and computational skills. The WhatsApp group puzzles, repeated incentivisation, and the informal nature of learning in and outside the classroom contributed to high engagement levels. These benefits were suitably channelised to foster a conducive learning environment for developing computational thinking subskills.

Most importantly, as students become familiar with techniques and skills, they think across time and space using History to study not just the past but to explore the present as well as the future.

9. REFERENCES AND CITATIONS

Appleton, James J. Christenson, S. L. et al. (2006). Measuring cognitive and psychological engagement: Validation of the Student Engagement Instrument. *Journal of School Psychology*, *44*(5), 427–445. Retrieved March 15, 2022 from https://doi.org/10.1016/j.jsp.2006.04.002

Czerkawski, B. C., & Lyman, E. W. (2015). Exploring Issues About Computational Thinking in Higher Education. *TechTrends*, *59*(2), 57–65. Retrieved March 12, 2022 from https://doi.org/10.1007/s11528-015-0840-3

Günbatar, M. S. (2019). *Computational thinking within the context of professional life : Change in CT skill from the viewpoint of teachers*. 2629–2652. Retrieved March 12, 2022 from https://link.springer.com/article/10.1007/s10639-019-09919-x

Henrique de Paula, B. et al. (2018). Playing Beowulf: Bridging computational thinking, arts and literature through game-makingNo Title. *International Journal of Child-Computer Interaction*, *16*, 39–46. Retrieved March 15, 2022 from https://www.sciencedirect.com/science/article/pii/S2212868917300247

Merino-Armero, J. M., González-Calero, J. A., Cózar-Gutiérrez, R., & Del Olmo-Muñoz, J. (2022). Unplugged Activities in Cross-Curricular Teaching: Effect on Sixth Graders' Computational Thinking and Learning Outcomes. *Multimodal Technologies and Interaction*, *6*(2). Retrieved March 15, 2022 from https://doi.org/10.3390/mti6020013

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B. (2012). Infusing computational thinking into the middle- and high-school curriculum. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, 22–27. Retrieved March 12, 2022 from https://doi.org/10.1145/2325296.2325306

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35. Retrieved March 9, 2022 from https://dl.acm.org/doi/fullHtml/10.1145/1118178.1118215

DOI: 10.34641/ctestem.2022.475

Design of an Evaluative Rubric for CT Integrated Curriculum in the Elementary Grades

Florence R. Sullivan, Lian Duan, Emrah Pektas
University of Massachusetts Amherst
florence@umass.edu, lduan@umass.edu, epektas@umass.edu

ABSTRACT

Despite the recent proliferation of research concerning integrating computational thinking (CT) into K-5th grade curriculum, there is little literature concerning how to evaluate the quality of CT integrated curricula, especially curricula integrating CT into language arts and social studies content areas. In this paper, we present a theoretically derived rubric for the evaluation of CT integrated curricula for grades K-5 across the curriculum (math, science, language arts, social studies). Our rubric is divided into two sections. The first section provides guidelines for identifying the integration type (disciplinary, multidisciplinary, interdisciplinary, or transdisciplinary). The second section presents six categories of evaluation that further subsume nine sub-categories. The principal categories of evaluation include the following: conceptual coherence, role of computational technology, assessment, use of multiple representations, play, and equity. We include the play category as an aspect of developmental appropriateness. Play is an important pedagogical approach for learning in the early grades. Our work takes place in the context of the Computer Science (CS) for All initiative in the United States which emphasizes the goal of improving racial and gender diversity in CS participation. Therefore, creating integrated lessons that address equity is important. Our paper describes rubric development from the theoretical perspectives that underlie the inclusion of each type, category, and sub-category. Our evaluative rubric can guide future efforts to integrate CT/CS into the elementary curricula. Researchers can utilize our rubric to evaluate and analyze CT-integrated curricula, and educators can benefit from using this rubric as a guideline for curriculum development.

KEYWORDS

Curriculum integration, computational thinking, elementary grades, evaluation rubric,

1. ELEMENTARY CT INTEGRATION

While introducing computational thinking (CT) in the elementary grades is not a particularly new idea (Bers, Flannery, Kazakoff, & Sullivan, 2014; Papert, 1981), it is, arguably, an increasingly important one. Many professions require facility with computers (Muro, Liu, Whiton & Kulkarni, 2017), and indeed 95% of children in the United States have access to a computational device and the internet in their own homes (National Center for Educational Statistics, 2019). Teaching children CT in the early grades is warranted. While this is so, the elementary school day has a full curriculum that leaves little room for introducing a stand-alone new topic such as computer science (Sherwood, et al., 2021). To introduce CT in elementary school an integrated approach is needed. Indeed, advocates of integrated curriculum believe that utilizing naturally overlapping areas of disciplines to integrate curriculum leads to higher student engagement and consequently higher achievement (Drake & Burns, 2004; Hinde, 2005; Vars, 1991). However, curricular integration is not a simple task. It requires attention on many levels. Here we present our work on the development of a CT-integration evaluation rubric for elementary curricula. The evaluation rubric is theoretically grounded and builds on prior work. Our evaluation rubric is unique in that no other comprehensive, elementary level CT-integration evaluation rubric exists. This rubric can be used both to evaluate existing curricula, or as a guide to curriculum development.

2. CT INTEGRATION APPROACHES

In a review of the literature, we have identified three basic approaches to integrating CT in the K12 curriculum as follows: general conceptual or practice overlap (Dong, Cateté, Jocius, Lytle, Barnes, Albert, Joshi, Robinson, & Andrews, 2019; Settle, Frank, Hansen, Spaltro, Jurisson, Rennert-May, & Wildeman, 2012); specific conceptual or practice overlap (Clark & Sengupta, 2020; Israel & Lash, 2020; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Weintrop, Behesti, Horn, Orton, Jona, Trouille, & Wilensky, 2016); and/or general content support (Waterman, Goldsmith & Pasquale, 2020). In terms of the first approach to integration, researchers have identified general practices and or concepts that are common to both computer science and other disciplines. For example, Dong et al., (2019) identified Pattern Recognition, Abstraction, Decomposition, and Algorithms (PRADA) as general concepts, that while foundational to computer science, are also found in many disciplines. The PRADA concepts can be used to approach problems in multiple fields. Similarly, Settle, et al. (2012), identified abstraction as a general concept, foundational to computer science work, and widespread among other disciplines.

The second approach to integration is to identify specific conceptual or practice overlaps between computational thinking ideas and other disciplines. This approach is typically focused on integrating CT into either math or science curricula. For example, both Clark and Sengupta (2020) and Sengupta, et al., (2013) identified modeling as a specific practice in science and computer science. Moreover, computational modeling is, at this point, an indispensable aspect of most scientific research. Israel and Lash (2020) identified three specific concepts in CT and math including sequencing, looping, and conditional logic. Meanwhile, Weintrop, et al., (2016) developed a comprehensive guide to the relationship of CT to the disciplines of math and science at the secondary level,

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

including four overarching categories and 22 practices that are specific to both CT and math and science.

Finally, the third approach focuses on general content support. Waterman, Goldsmith and Pasquale (2020) pre-identified three forms of integration of CT into the science curriculum: exist, enhance, extend. Working with the third-grade science curriculum, these researchers identified science topics where CT naturally existed as part of the inquiry activity, places where CT could enhance the learning of the topic, and places where CT could extend the learning of the science topic.

These various approaches to CT integration are valuable for curriculum developers and teachers. However, they do not, in and of themselves, speak directly to the quality of a particular CT integrated curriculum. Therefore, we have worked to develop a CT integration evaluation rubric for the elementary grades. Our rubric addresses issues of quality, developmental appropriateness, and equity. In the balance of this paper, we describe our development process, and we provide the theoretical grounding for the presence of each category of evaluation. Our goal in undertaking this work is to furnish the CS education research community with a useful tool for making important curricular decisions regarding selecting or developing a high-quality CT-integrated curriculum for the elementary grades.

3. RUBRIC DEVELOPMENT PROCESS

Our rubric development process proceeds from the literature regarding curriculum integration. Some papers have focused on identifying different types of integration– which we discuss in section 4 below. Other papers have focused on, or identified, important elements of quality that should be considered when working to integrate two or more disciplines – which we discuss in section 5 below. These quality indicators include conceptual coherence, the role of technology, assessment, and the use of multiple representations. We have included two other quality indicators that we believe are important and which contribute to the comprehensiveness of our rubric: play and equity. We include a focus on play due to the importance of play as a pedagogical approach in the early grades (National Association for the Education of Young Children, 2020). We include a focus on equity because in our United States context, there is a strong focus on improving the diversity of individuals who participate in CS including those from societally oppressed racial groups. Therefore, developing CS curricula that addresses issues of equity is important.

4. TYPES OF INTEGRATION

Various approaches to curricular integration have been posited over the years (see Davison, Miller & Metheny, 1995; Fogarty, 1991; Vars, 1991). Common to the approaches is the goal of finding overlapping connections among disciplines, such that integration is sensible. Such integration might occur through various mechanisms of overlap, for example, content integration, thematic integration, process integration, skill integration and correlational integration (Davison, et al., 1995; Fogarty, 1991; Vars, 1991). More recently, researchers have developed an integration model that includes three approaches and implies a fourth. The three approaches first discussed by Drake and Burns (2004) and later elaborated upon by Vasquez, Comer & Sneider (2013), include the following multidisciplinary, interdisciplinary, and transdisciplinary. The fourth, implied aspect is disciplinary – and this element is a part of Vasquez, Comer & Sneider's (2013) delineation of the types of integration. We have adopted their approach for our rubric. Therefore, our rubric has four types of integration: disciplinary, multidisciplinary, interdisciplinary and transdisciplinary.

We include the disciplinary category as some approaches to integration acknowledge the importance of developing specific disciplinary knowledge prior to engaging in multi, inter, or transdisciplinary learning (Kiray, 2012). Essentially, a unit might include some disciplinary learning prior to introducing its connection to another discipline. The second type of integration, multidisciplinary, refers to lessons or units where two disciplines are united by a common theme, but where the goals of the lesson for each discipline are not interconnected or interdependent. An example of a multidisciplinary approach to integrating CT into the curriculum would be selecting a particular theme, such as "plants" and then teaching about plants (e.g., the plant life cycle) using computational media, for example, have students create an animation of the plant life cycle from seed to flower using Scratch. The third type of integration is interdisciplinary. In this approach the two disciplines are conceptually connected, in other words a concept, common to both disciplines is at the heart of the lesson; and the learning goals for each discipline are interconnected and interdependent. An example of integrating CT with this approach is to identify a concept, such as "precision." This concept is important in both computer science and in learning how to write procedurally in English Language Arts, for example, writing precise instructions. Finally, there is the transdisciplinary type of integration. In this approach, the focus is on approaching a real-world problem from multiple disciplinary lenses. An example of integrating CT using this approach would be to identify a community problem, for example the presence of large potholes in the streets, and then develop a plan for solving the problem from various lenses, including sociological (survey the community to discover thoughts about the problem), English Language Arts (write up the results of the survey) and computer science (create an application using GPS technology that allow people to automatically flag the location of a pothole). It is important to recognize that each of these types of integration (multidisciplinary, interdisciplinary, and transdisciplinary) are equally valid and equally useful. The approach selected should be driven by the overall goals of a given lesson or unit (Kiray, 2012).

5. QUALITY INDICATORS

Here in section 5, we will describe the quality indicators, including discussing their roots in the literature. We have developed a four-point qualitative evaluation system including the following assessments: poor, fair, good, excellent. These assessments use a graduated presence-

absence evaluative approach. For example, if a quality indicator is judged as poor, it is judged so due to the wholesale absence of the indicator. The rest of the assessment levels move in a graduated way towards the full presence of the quality indicator. Due to space limitations, we are not able to provide a detailed description of each element of the four-point evaluative rubric for each quality indicator. However, we plan to publish the full rubric at a later date. Here we provide the theoretical underpinnings and rationale for the rubric.

5.1 Conceptual Coherence

The first quality indicator is conceptual coherence (Roehrig, Dare, Ring-Whalen & Wieselmann, 2021). Coherence can be achieved through scanning curricular standards to find synergistic integration points, assembling these points to ensure horizontal and vertical progression throughout the school year and across the grades, designing learning activities to achieve the learning objectives in the integrated subject areas, and aligning standards and learning goals and activities with assessments (Drake & Burns, 2004; Case, 1994). While conceptual coherence may be evaluated across varying timescales, for the purpose of this evaluative rubric, coherence concerns the relationship among concepts introduced in a lesson. For example, how are the concepts sequenced and linked to one another? How do the concepts work together to build a picture of the topic of interest? How interrelated are the concepts?

We evaluate this indicator on two levels: (1) the coherence of CT concepts throughout the lesson; and (2) the coherence of the CT concepts with the target domain concepts in the lesson. In terms of our rubric, CT concept coherence refers to CT concepts being introduced in a clear, meaningful order. For example, in a lesson that introduces the CT concepts of algorithms and debugging, we would expect to see the concept of algorithm introduced first, then the concept of debugging. In terms of coherence across two disciplines, we would be looking for the overlap and connection among concepts. For example, in a third-grade lesson that is introducing the concept of algorithms within the context of an English Language Arts lesson, specifically a lesson on story structure (e.g., first, then, next, last) we would look for how the lesson connects the idea of an algorithm as a specific sequence of steps to the idea of story structure also as a specific sequence. A good example of how to connect these ideas is to have children use the Scratch technology to create a short, animated story in Scratch that uses the simple story structure, first, then, next, last. Indeed, Burke & Kafai (2010) have demonstrated that a similar technique has been successful with teaching older children about both coding and writing stories using Scratch.

5.2 Role of Computational Technology

The second quality indicator concerns the role of computational technology in the CT-integrated lesson. By the role of technology, we mean the way the technology is used to support student learning, with a special focus on the extent to which the technology supports learning in all of the disciplinary topics included in the lesson. For example, as noted above, Burke & Kafai (2010), utilized the Scratch technology to examine student learning of coding, as well as their learning related to creative writing. We (Authors, 2021) found similar support for student learning of coding and elements of narrative when Scratch was integrated into a fourth-grade classroom. A primary reason why Scratch appears to be a suitable technology for teaching Language Arts (for example, narrative elements in storytelling), is the design of the technology itself. Scratch is developed using a theatrical metaphor of "the stage" for which one creates or selects a background, selects or creates characters (termed "sprites" in the Scratch software), and then develops either some sort of animation, an interactive story or an interactive game. Aris is another computational media that uses a narrative metaphor to engage students in game design and storytelling across any number of disciplines (Litts, Lewis & Mortensen, 2020).

Other computational technologies that support CT-integrated learning include those that support both programming and modeling activities like NetLogo (Wilensky, 1999), AgentSheets (Repenning, 1993) and CTsim (Sengupta et al., 2013). In these technologies, students use code to set the parameters to run simulations and create models. These technologies can also be used to create games that integrate learning in CT, science and math (Clark & Sengupta, 2020), and potentially appeal to youth interests in doing so. The appropriateness of a computational technology for supporting learning across the integrated disciplines is key here.

5.3 Assessment

Our third quality indicator is assessment. Assessment plays a key part in any type of learning for students and educators alike. Through well-designed assessments, aligned with learning goals and with clear criteria, students not only gain information on what they know and where they need improvement, but also establish trust in teachers (Guskey, 2003). Teachers utilize assessments to identify troubled spots, understand the nature of students' struggles, and examine and adjust their teaching methods (Guskey, 2003). Assessments of CT-integrated lessons must measure both CT and domain knowledge of the integrated academic subjects. For assessment to reflect both CT and domain knowledge in subject matter, Drake and Burns (2004) suggest pulling overlapping standards apart to record separately students' progress in each subject of the integrated curriculum.

Moreover, the use of multimodal assessments is key, especially where equity is concerned (Burke & Kafai, 2010). Multimodal assessments include but are not limited to software metrics, audio and video recordings, and observation notes. These assessments allow students to demonstrate their understanding and competency applying concepts and skills and express their dispositions and attitudes towards computational thinking (Burke & Kafai 2010; Tang, Yin, Lin, Hadad, & Zhai 2020). Based on these understandings, we include in our criteria for assessment the following subcategories: alignment with integrated learning objectives and multimodality.

5.4 Multiple Representations

The fourth quality indicator concerns the use of multiple representations in the lesson, and specifically representations that are relevant to the disciplines being integrated. Others have argued for the importance of developing representational competence (Ainsworth, 2006; Kozma & Russell, 2005) as an important aspect of learning in a specific discipline, like science. Representational competence refers to the ability to be able to read and understand specific modal representations. This is important because discipline specific representations encode the social and cognitive affordances in the material features of the representation (Kozma, 2003). Therefore, understanding disciplinary representations is an important element of understanding in the discipline.

Meanwhile, researchers have begun to identify representations that bridge disciplines, and would, therefore, be very useful in helping students learn in an interdisciplinary fashion. Sengupta et al., (2013) identified representations that result from computational modeling activities as specific to the fields of computer science, math, and science. These models are typically abstract representations of a phenomena. In this case, the concept that can be taught in an interdisciplinary fashion via the development of the representation is abstraction. This work was followed up by Clark and Sengupta (2020) who pioneered the use of modeling in disciplinary integrated gaming (DIGs) environments. They argue that "…the design of DIGs focuses on engaging students more deeply in specific representational practices of developing, interpreting, manipulating, and translating across specific disciplinary model types" (p.330). In our evaluative rubric, we analyze the existence of disciplinary representations in the lesson, and specifically, the degree to which specific representations are utilized and how they align with the CT-integrated learning goals of the lesson.

5.5 Play

Our fifth quality indicator is the role of play in the lesson. Play is an important component of learning for young children as the National Association for the Education of Young Children (NAEYC) (2020) argues "Play promotes joyful learning that fosters self-regulation, language, cognitive and social competencies as well as content knowledge across disciplines. Play is essential for all children, birth through age 8" (p. 9). They further argue that play is a major developmentally appropriate approach for preschool and early elementary and a universal phenomenon across all cultures (NAEYC, 2020). Since our evaluative rubric is designed for evaluating elementary level (Kindergarten – Grade Five) CT-integrated curriculum, developmental appropriateness is one of the theoretical lenses through which we developed our rubric.

What makes an activity play? How is play defined? Vygotsky (1978) argues that there are four criteria that make up play including: (1) play is fun, (2) play has rules (explicit or implicit) (3) play includes imaginary situations (explicit or implicit), and (4) play has a purpose. We included the latter three elements in our evaluative rubric. We excluded the criteria of fun due to the thoroughly subjective nature of the concept. Further, we divide our analysis of play as an aspect of the CT-integrated curriculum into two sub-categories: playful activities and games. We distinguish among these elements as Vygotsky (1978) did to provide a level of precision in analysis. For example, in playful activities, the rules are implied and the imaginary situation is explicit. Whereas in a game, the rules are explicit and the imaginary situation is implied. Therefore, our evaluative rubric first distinguishes between playful activities versus games that might be used to present the curriculum. Then, we evaluate the degree to which the elements of play (has rules, has an imaginary situation, has a purpose) are discernible for students and support learning.

5.6 Equity

The sixth and final quality indicator is equity. Due to the long history of the oppression of people of color in the United States (Kendi, 2016), we are specifically interested in addressing racial equity in our evaluative rubric. This is not meant to underplay the importance, and indeed, necessity of addressing gender equity, but due to the context in which we have developed our evaluative rubric, our current focus is on racial equity. Here, we draw most notably upon the work of Muhammad (2020), who developed a four-layered equity framework named the Historically Responsive Literacy (HRL) Framework. This framework includes the following elements: (1) identity development, (2) skill development, (3) intellectual development, and (4) Criticality. While Muhammad's (2020) framework focuses on the teaching of literacy, we adopt it here due to the relevance of the elements to supporting students in developing computational thinking. Muhammad (2020) argues that students have the potential for success when their identity such as culture, gender, and race is incorporated in the curriculum and affirmed. This notion is affirmed by the work of Cheryan, Plaut, Davies & Steele (2009) who demonstrated how cultural elements of computer science learning environments left women feeling excluded. Without seeing themselves and their interests reflected in computer science learning environments, women were less interested in pursuing the field.

Muhammad (2020) defines skills as "competence, ability, and expertise based on what educators deem to be important for student learning in each content area" (p. 85). She argues that skills should be taught in a context that provides social, emotional, or intellectual relevance to students, and they should be given opportunities to put the skills learned into practice. Muhammad (2020) defines intellect as "what we learn or understand about various topics, concepts, and paradigms" (p. 104). In other words, as learning takes place, one asks, "What am I becoming smarter about?" According to Muhammad, intellect also holds the meaning of applying the knowledge learned into action. Finally, Muhammad (2020) differentiates between "c" critical and "C" Critical. For her, while critical means to think deeply about something, Criticality is related to power, power dynamics, entitlement, oppression, and equity. She defines Criticality as the "…capacity to read, write, and think in ways of understanding power, privilege, social justice, and oppression, particularly for populations

who have been historically marginalized in the world" (p. 120). We adopt Muhammad's four-layer approach in evaluating CT-integrated lessons for their attention to equity. In other words, we evaluate the degree to which children can see themselves in the lesson (identity), learn and practice skills in the lesson (skills), become knowledgeable about computer science, what it is and how it fits, broadly, into our world (intellect) and engage with aspects of societal oppression (Critically). This last element may be easier to accomplish through CT-integrated lessons in Language Arts and Social Studies.

6 RUBRIC SCALE AND APPLICATION

The evaluation rubric has 14 items, including the quality indicators described above and their subcategories. Seven items might not apply to some lessons due to their absence, including 1) role of computational technology, 2) playful activity rules, 3) playful activity purpose, 4) playful activity imaginary situation, 5) game purpose, 6) game rules, and 7) game imaginary situation. The 14 items in the rubric have an equal weight of 4 points each. Four rating categories are utilized as has been recommended in the literature (Stone, 2003), each of which corresponds to a score ranging from 1 to 4 points.

Our application of the rubric started with a testing evaluation of 11 lessons. Two research assistants rated the lessons independently and compared the results afterward. Disagreements were resolved through discussions to clarify the criteria for the four grading levels and the presence and absence of certain curricular elements, such as play or technological tools. For example, to differentiate "good" (3 points) from "excellent" (4 points) for the category "skill," the defining element was decided to be "explicit teaching or discussion of how the skills reflect the professional practices of computer scientists or professionals of other disciplines." The evaluators then followed the agreed-upon research notes detailing these clarifications to grade the lessons. We make judgments based on our experience as researchers and veteran K-12 educators. We utilized simple scoring and converted the score into a percentage as the overall rating. Since the rubric is over ten pages in length, we are unable to provide an evaluative example here.

7 CONCLUSION

Here we have presented the design of our evaluative rubric for CT-integrated lessons in the elementary grades. This evaluative rubric is an important adjunct to the tools CS educators and researchers have available to them for selecting and/or creating quality, CT-integrated curriculum for elementary schools. As noted earlier, due to time constraints, it is most likely that the discipline of computer science will need to be integrated across the curriculum at the elementary level, if it is to be taught at all. We have endeavored to design a comprehensive evaluative rubric. This rubric not only attends to types of integration (multidisciplinary, interdisciplinary, and transdisciplinary approaches), but also to what to look for in terms of quality (conceptual coherence, the role of technology, assessment and the use of multiple representations). Finally, we have attended to both the developmental appropriateness of pedagogical approaches, as well as issues of equity in curriculum development. Because of the integrated nature of including CS in the elementary curriculum, a rubric such as ours is an important and much needed tool.

Our future work includes applying this rubric to a set of 115 lessons created for kindergarten through 5th grade, both to validate the use of the rubric, as well as evaluate the quality of the curriculum.

8 REFERENCES

Ainsworth, S. (2006). Deft: A conceptual framework for considering learning with multiple representations, *Learning and Instruction, 16,* 183-198.

Authors, (2021).

Bers, M.U., Flannery, L. Kazakoff, E.R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72,* 145-157.

Burke, Q., & Kafai, Y. B. (2010, June). Programming & storytelling: opportunities for learning about coding & composition. *In Proceedings of the 9th international conference on interaction design and children* (pp. 348-351).

Case, R. (1994). Our crude handling of educational reforms: The case of curricular integration. *Canadian Journal of Education, 19*(1), 80-93.

Clark, D.B., & Sengupta, P. (2020) Reconceptualizing games for integrating computational thinking and science as practice: collaborative agent-based disciplinarily-integrated games, *Interactive Learning Environments, 28*(3), 328-346, DOI: 10.1080/10494820.2019.1636071

Cheryan, S., Plaut, V.C., Davies, P.G., & Steele, C.M. (2009). Ambient belonging: How stereotypical cues impact gender participation in computer science. *Journal of Personality and Social Psychology, 97*(6), 1045-1060.

Davison, D.M., Miller, K.W., & Metheny, D.L. (1995). What does integration of science and mathematics really mean? *School Science and Mathematics, 95*(5), 226-230.

Dong, Y., Cateté, V., Jocius, R., Lytle, N.,Barnes, T. Albert, J., Joshi, D., Robinson, R., & Andrews, A., (2019). PRADA: A practical model for integrating computational thinking in K-12 education. *In proceedings of the annual meeting of the ACM SIGCSE '19, February 27–March 2, 2019, Minneapolis, MN, USA.*

Drake, S.M. & Burns, R.C. (2004). Meeting standards through integrated curriculum. *Association for Supervision and Curriculum Development.* Alexandria, VA

Fogarty, R. (1991). Ten ways to integrate curriculum. *Educational leadership, 49*(2), 61-65.

Guskey, T. R. (2003). How classroom assessments improve learning. *Educational Leadership, 60*(5), 6-11.

Hinde, E.T. (2005). Revisiting curriculum integration: A fresh look at an old idea. *The Social Studies*, *96*(3), 105-111. DOI 10.3200/TSSS.96.3.105-111

Israel, M., & Lash, T. (2020). From classroom lessons to exploratory learning progressions: mathematics+ computational thinking. *Interactive Learning Environments 28*(3), 362–382.

Israel, M., Pearson, J.N., Tapia, T., Wherfel, Q.M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education, 82*, 263-279.

Kendi, I.X. (2016). *Stamped from the beginning: The definitive history of racist ideas in America.* Nation Books.

Kiray, S. A. (2012). A new model for the integration of science and mathematics: The balance model. *Energy Education Science and Technology Part B: Social and Educational Studies*, *4*(3), 1181–1196.

Kozma, R. (2003). The material features of multiple representations and their cognitive and social affordances for science understanding. *Learning and Instruction*, *13*, 205-226.

Kozma, R., & Russell, J. (2005). Students becoming chemists: developing representational competence. In J. K. Gilbert (Ed.), *Visualizations in science education* (pp. 121-146). Dordrecht, The Netherlands: Springer.

Muro, M., Liu, S., Whiton, J. & Kulkari, S. (2017). Digitalization and the American workforce. *The Brookings Institute.* Online at https://www.brookings.edu/research/digitalization-and-the-american-workforce/

National Association for the Education of Young Children, (2020). Developmentally appropriate practice: *A position statement of the National Association for the Education of Young Children.* Online at https://www.naeyc.org/sites/default/files/globally-shared/downloads/PDFs/resources/position-statements/dap-statement_0.pdf

Repenning, A. (1993). Agentsheets: A tool for building domain-oriented visual programming. *Conference on Human Factors in Computing Systems.* 142–143.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: a theoretical framework. *Education and Information Technologies*, *18*, 351e380. http://dx.doi.org/10.1007/s10639-012-9240-.

Settle, A., Franke, B., Hansen, R., Spaltro, F., Jurisson, C., Rennert-May, C., & Wildeman, B., (2012). Infusing computational thinking into the middle- and high-school curriculum. *In proceedings of the annual meeting of the ACM ITiCSE'12,* July 3–5, 2012, Haifa, Israel.

Sherwood, H., Martin, W., Rivera-Cash, E. Yan, W., Adair, A., Pierce, M. Liu, R., Fancsali, C., Israel, M. (2021). Diverse approaches to school-wide computational thinking integration at the elementary grades: A cross-case analysis. *In Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21), March 13–20, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages.

https://doi.org/10.1145/3408877.3432379

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, *148*, 103798.

U.S. Department of Education, National Center for Educational Statistics (2021). The condition of education 2021 *(NCES 2021-144).* Children's internet access at home. Online at https://nces.ed.gov/fastfacts/display.asp?id=46

Vars, G.F. (1991). Integrated curriculum in historical perspective. *Educational Leadership*, *10*, 14-15.

Vasquez, J. A., Comer, M., & Sneider, C. (2013). *STEM lesson essentials, grades 3–8: Integrating science, technology, engineering, and mathematics.* Portsmouth, NH: Heineman.

Waterman, K.P., Goldsmith, L., & Pasquale, M. (2020). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology 29* (1), 53–64.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U., (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*, 127–147. DOI 10.1007/s10956-015-9581-5.

Wilensky, U. (1999). NetLogo. *Center for Connected Learning and Computer-Based Modelling (http://ccl.northwestern.edu/netlogo).* Northwestern University, Evanston, IL.

DOI: 10.34641/ctestem.2022.476

Scaffolded programming projects to promote computational thinking

Victor KOLESZAR[1], Alar URRUTICOECHEA[1], Andrés OLIVERI[1], María del Rosario SCHUNK[1], Graciela OYHENARD[1]
[1]Computational Thinking Unit, Ceibal, Uruguay
vkoleszar@ceibal.edu.uy, aurruticoechea@ceibal.edu.uy, aoliveri@ceibal.edu.uy, mschunk@ceibal.edu.uy, goyhenard@ceibal.edu.uy

ABSTRACT
In Uruguay, Plan Ceibal drives the complex task to impulse computational thinking in public schools. The CT framework used by the organization is to introduce computer science from primary and secondary education, with an approach focused on solving problems and coding as a language, and with the intention of taking advantage of the potential of computational thinking. In order to educate users and creators of technology. In 2021 the Computational Thinking program of Plan Ceibal impacted nearly 40 thousand students and teachers, this represents about 30% of the enrollment for K 4 to 6 courses (9 to 11 years old) of the public elementary school. This study explored the impact of the implementation of scaffolded programming projects and final evaluation, in a subset of elementary schools groups. Preliminary results suggest a good adoption of the program and high participation of students and teachers registered through the learning management system (LMS) platform. In addition, the students who had more active participation in the classes had significantly higher performances in the programming tests. Some differences were observed in favor of girls. Results are discussed in relation to the pedagogical characteristics of the program.

KEYWORDS
Computational Thinking, Learning and Teaching, K-12

1. INTRODUCTION
The increase and ease of use of different technological tools in all areas of life has generated the need to integrate them in the classroom in order for students to acquire the necessary skills that allow them to face the difficulties or challenges that arise from this practice (Goyeneche et al., 2021). In response to this new scenario, different countries are developing and implementing educational programs that can meet the technological needs of students, with the intention of reducing the digital divides that some socio-cultural sectors are unable to access technology, ensuring equitable access and critical uses. Within this framework, the concept of Computational Thinking (CT) as a set of competencies for the expression and resolution of problems using the logic of programming and the power of computers is gaining strength, repositioning computer sciences. In Uruguay, the educational innovation center with digital technologies Plan Ceibal, launched in 2017 a pilot program of introduction to computer science with classroom intervention. The Computational Thinking program grows from 50 groups of students at its inception to 1768 groups of students in 2021 (approximately 35 thousand students in primary education) (Koleszar et al., 2021).

Likewise, the program continues to be optional, but classes are held during curricular hours. Teachers who enroll their groups incorporate a remote computational thinking teacher into the classroom work.

The objective of this research is to present the results of the evaluation carried out on the children who have participated in The Drawing Machine project during the implementation of the CT program in 2021.

2. PEDAGOGICAL MODEL
The pedagogical model comprises a set of initiatives and materials to cover different aspects of classroom and teacher work. They are composed of a didactic sequence of learning activities, a training course for teachers, materials and resources for the virtual platform CREA (LMS), and a final evaluation for students.

The didactic sequences of Plan Ceibal's CT program are based on the following aspects: the importance of designing and creating activities that motivate and generate better learning experiences for students (Resnick & Silverman, 2005). In this sense, the sequences combine directed, guided and exploratory activities that give structure to the projects. The teaching of programming is used as the main approach to promote computational thinking (Scherer et al., 2019). Activities are designed with progression: use-modify-create (Lee et al., 2011), with incremental program cycles and inquiry learning methodologies (Furmann, 2016). Projects should be motivating, promote collaborative work, teamwork, and significant play (Resnick, 2014)

The program is organized in three levels (4th, 5th and 6th grade), in which the contents and competencies of computational thinking are covered in an incremental and sequential manner. For each level, several didactic sequences were developed, which in an interdisciplinary way integrate Computational Thinking with other areas of knowledge. The design of these sequences is carried out in collaboration with the Argentinian Sadosky Foundation. The sequences are organized in projects that propose the design and construction of devices or programs with a theme of choice and structure of the contents and practice. The aim is that teachers can make didactic transpositions that allow students to solve problems related to Mathematics, Social or Natural Sciences, Language, etc., or associated to real life situations, through the skills and competences developed by computational thinking.

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

Each project is presented as a complex challenge to be solved in a period of approximately 6 to 8 weeks, with a dedication of 45 to 60 minutes per week. The teaching approach of the sequences aims at high-level thinking, where a problem or project is presented, and students have the opportunity to explore solutions, to transfer different concepts, to create programs or devices, in a balance of guided activities and peer-to-peer work space. The focus is on practices and concepts rather than tools. The sequences are open in relation to the subject matter so that classroom teachers can link it with the contents of the program they are dealing with; flexible in terms of the complexity of the programming so that teachers can adjust the requirements according to the experience of the group of students and creative in that it places students as designers and creators of stories, video games, simulators, robotic devices, etc.

In each videoconference there are initial activities that organize the exchange so that students can tell the remote teacher what they have done between videoconferences; development activities that allow them to advance in the proposal; and closing and reflection activities that are fundamental to recover moments that have been observed during the development and to promote metacognition.

CT program seeks to promote an inclusive educational experience that promotes gender equity. In order to do that, classroom and remote teachers are attentive to constantly denaturalize the bias of computer science and programming as an exclusive male task.

2.1 The Drawing Machine

This paper deals with the results achieved from the project called the drawing machine (TDM), the first project carried out by the students at the beginning of the program, in 2021. This proposal consists of the design and programming in Scratch of a machine capable of drawing from the interaction with the user. Figure 1 summarizes the main elements of the 8 stages of the project.

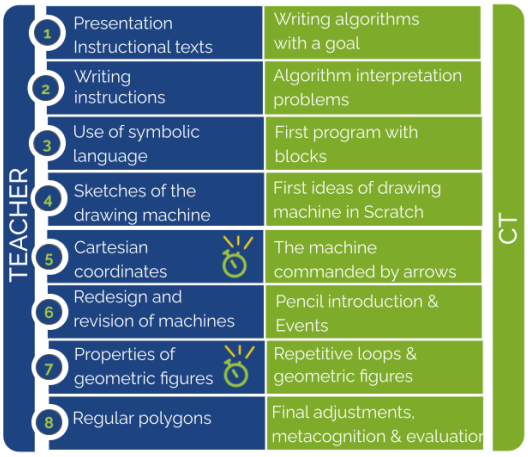


Figure 1. Synthesis of The Drawing Machine sequence

This didactic sequence proposes a series of activities for students to go through computational practices (decompose and plan, abstract and modularize, test and debug, reuse and reinvent) and programming concepts (algorithms, programs, instructions, events and repetitions). Figure 2 illustrates an example of the class guide for the CT remote teacher.

Three moments of the class are highlighted. The introduction or warm-up where the topic is introduced or the previous work is recovered; the development of the central activity with activities that advance in the project; and finally the closing with triggering questions to evaluate the process and final conceptualizations.

CT video conference 4↓
The Drawing Machine in Scratch

Challenge task
Draw the machine in Scratch.
Create a new object with the Scratch editor that represents the drawing machine.

1. Warm up (15 min)
Who was interpreting the code when they made the house that someone else described? And when did they make the drawings with the arrow code? And when was Coty scheduled?
The students recover from the work the notions built between all of the algorithms, program and language. It is anticipated that at this stage they will begin to design and program the Drawing Machine using a new block programming language: Scratch.

2. Development Designing the drawing machine (20/25 min)
The challenge is presented and a very brief presentation of Scratch, its workspaces and the drawing editor is presented. Time is set aside for exploration and resolution of the challenge.

Suggestions:
Design: Students must clearly identify which part of the machine is going to make the drawings.

3. Closure (10 minutos)
Was it easy to adapt the designs? Is the editor similar to another one they have used? What advice would you give to someone drawing an object for the first time in Scratch?
Students share their designs, explain what difficulties they encountered and record design tips or tricks in Scratch. They analyze what they have done by comparing it with other drawing experiences on the computer. By way of conclusion, it is highlighted that graphic editors usually have similar logics, iconographies and tools, and the importance of understanding the options offered by the program is pointed

Registration
Registration of progress in the CREA forum (LMS)

La Yapa: Home proposals
Make a new version of your drawing machine using the tips and techniques we share in the videoconference.

Figure 2. Example of a CT script class

3. METHODS & MATERIAL

An ad hoc questionnaire on concepts and practices worked on in TDM was used for the evaluation. It consists of 13 questions divided as follows: 8 multiple choice, 3 multiple response and 2 true or false. Figure 3 shows item 13 associated with repetitive structures. Internal consistency analysis showed acceptable results.

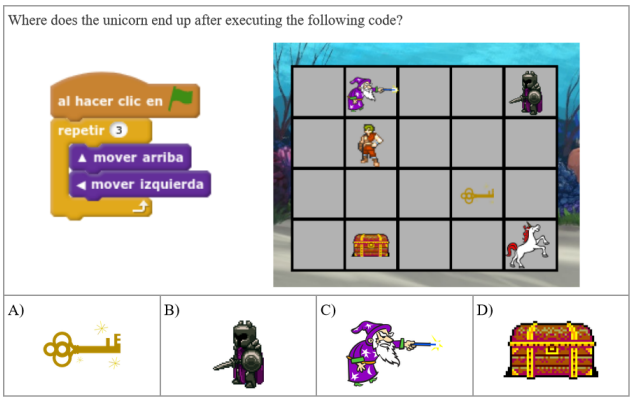


Figure 3. Evaluation task example

This evaluation was carried out through CREA (LMS) used on a daily basis, and in a classroom context. From this platform, data were obtained to identify the children, such as: date of birth, gender, school grade, socio-cultural level (five quintiles), area (urban/rural) and region of the country (department). Data on the use of the CREA platform on

Computational Thinking materials were also considered, specifically, number of entries, readings, homework submissions and forum comments. Four groups of use frequency were created by quartiles, taking into account these data: low use, medium low, medium high, and high use, in order to compare differences in performance.

4. RESULTS

4.1 Participants

Of the universe of students participating in the CT Ceibal program, a total of 3773 students participated in this evaluation, with a mean age of 133 months and a standard deviation of 11.70 months, at the time of the evaluation, from grades 4, 5 and 6 of elementary school. About gender, 50.1% are girls and 49.9% boys. Of the schools, 96% are urban, 3.7% are rural and 0.3% belong to another country, so there is no categorization (urban/rural).

Of the total number of children, 26.2% attend 4th grade, 32.5% attend 5th grade, 35.5% attend 6th grade and 5.8% attend multilevel shared classrooms.

Taking into account the sociocultural level to which the schools belong 576 children attend schools of low sociocultural level, 716 attend schools of medium-low, 585 attend schools of medium level, 728 attend schools of medium-high, 1155 attend schools of high sociocultural level (see Figure 1), 11 students are from a Uruguayan managed school but in Paraguay so it does not have socio cultural categorization and 2 students have missing data.

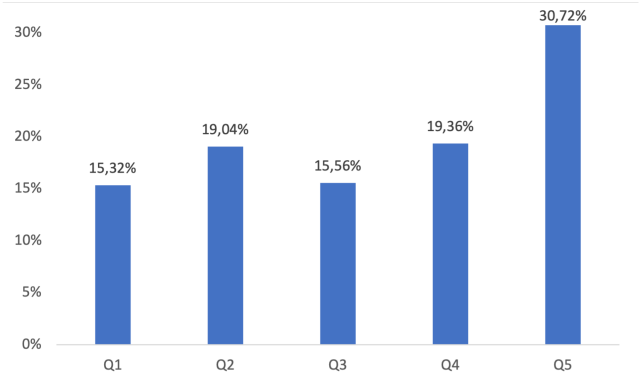


Figure 4. Students distribution by sociocultural quintile.

4.2 Descriptives and findings

This section shows the results: first presenting a descriptive analysis of the scores and then the results of the comparison of means by gender (t-test), sociocultural level, grade, and frequency of use of CREA (ANOVAs).

The mean score obtained in the assessment was 6.88 (SD = 2.81), with scores across the entire possible range of points (0 to 13) (see Figure 5).

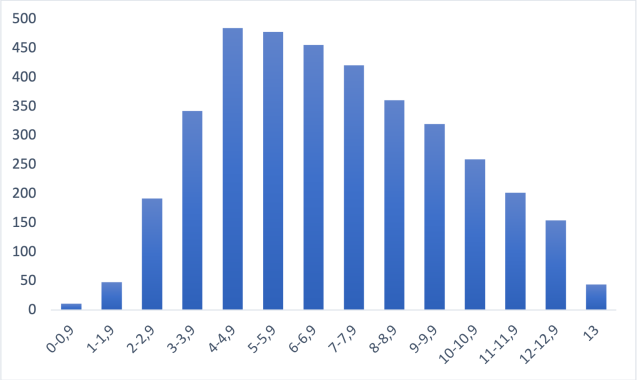


Figure 5. Distribution of students total scores

Table 1 shows statistically significant differences in the scores in favor of the female gender.

Table 1. Mean comparison by gender

	t	df	p
Score	-2.34	3771	.02

From the Analysis of Variance (ANOVA) taking into account the sociocultural quintile of the schools attended by the children (Table 2), it is noted that there are statistically significant differences in the performances obtained. The post-hoc analysis shows that the statistically significant differences are between the groups of low, medium-low and medium socio-cultural levels (Q1, Q2 and Q3) and the groups of medium-high and high socio-cultural levels (Q4 and Q5). Within these two groups of quintiles there are no statistically significant differences.

Table 2. Comparison of means by sociocultural quintile

Variable	df	F	p
Quintile	4	11.07	< .00

Considering the school grade to which the students belong, it is observed that there are significant differences in the scores (Table 3). Considering the post-hoc analysis, it is observed that the differences are between the sixth grade and the rest, while there are no statistically significant differences between the fourth and fifth grades.

Table 3. Means comparison by grade

Variable	df	F	p
Quintile	2	29.43	< .00

Finally, taking into account the use of computational thinking activities on the platform CREA, there are statistically significant differences in the performance between the groups created by the quartiles of frequency use ($F(3, 942)=91.43$; $p<0.001$). The post-hoc analysis yields differences between all the groups, always in favor of the higher use grouping, asi shown in Figure 6. A two-way ANOVA was realized to reject the possible effect of confounding independent variables ($F(24, 23)=.68$; $p=.88$).

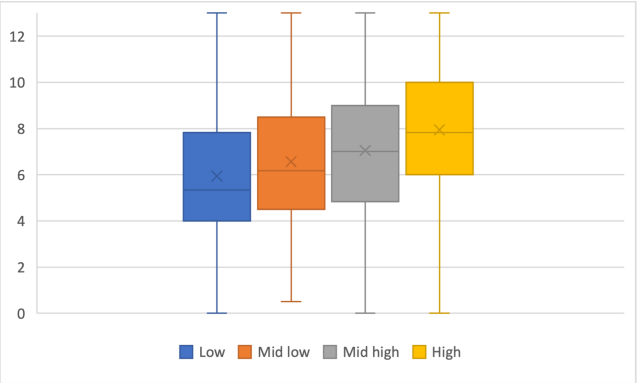


Figure 6. Performance according to frequency of use of CT activities

5. CONCLUSION AND DISCUSSION

Considering the results obtained in this research, it can be concluded that there are effects on the assessment scores:

1. By gender, girls obtain higher scores than boys.
2. By sociocultural level, there are two possible groupings taking into account the scores obtained by sociocultural level, the first being the first three levels (Q1, Q2 and Q3) and the second the two highest levels (Q4 and Q5). Within these groups there are no statistically significant differences, but between groups there are.
3. By grade, although there are differences between the means of all grades, only the difference between the 4th and 6th grades and between the 5th and 6th grades are statistically significant. There are no statistically significant differences between 4th and 5th grades.
4. Frequency of use of the CREA platform. The higher the use of the platform shows higher mean performances in the test, this happens from the lowest level (low level of use) to the highest (high level of use), happening progressively in the rest of the levels.

The statistically significant differences found in favor of women go hand in hand with what has been proposed by some authors who state that at the educational level, in general, women have better performance, (Driessen & van Langen, 2013), in turn, this contradicts other authors who found no statistically significant differences in programming skills by gender (Price & Price-Mohr, 2021). The Analysis of Variance (ANOVA) by sociocultural level provides an expected result, in which children from more favorable sociocultural contexts scored better on the assessment compared to those from less favorable sociocultural contexts (Liu et al., 2020). The results found on the use of CREA and the scores go hand in hand with those found by other authors, in which a relationship is found between the use of LMS and academic performance (Kim, 2017). In this sense, it is a good sign that participation in the program could promote computational thinking and programming skills.

6. FURTHER WORK

In order to obtain more evidence to help us distinguish causal effects of the CT program, it is being planned to conduct new interventions with pre-post experimental design.

7. REFERENCES

Driessen, G., & van Langen, A. (2013). Gender differences in primary and secondary education: Are girls really outperforming boys? *International Review of Education*, *59*(1), 67-86. https://doi.org/10.1007/s11159-013-9352-6

Furmann, M. (2016). *Educar mentes curiosas: La formación del pensamiento científico y tecnológico en la infancia: documento básico: XI Foro Latinoamericano de Educación La construcción del pensamiento científico y tecnológico en los niños de 3 a 8 años* (Primera edición). Fundación Santillana.

Goyeneche, J. J., Pereiro, E., Koleszar, V., Angeriz, E., Pérez, M., & Urruticoechea, A. (2021). Pensamiento computacional, proceso de creación de un videojuego de medida estandarizada. *International Journal of Developmental and Educational Psychology*, *1*, 25-32.

Kim, D. (2017). *The impact of learning management systems on academic performance: Virtual Competency and student Involvement*. *17*(2), 13.

Koleszar, V., Pérez Spagnolo, A., & Pereiro, E. (2021). *Pensamiento computacional en educación primaria: El caso de Uruguay*. Jornadas Argentinas de Didáctica de las Ciencias de la Computación, Buenos Aires, Argentina.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*(1), 32-37. https://doi.org/10.1145/1929887.1929902

Liu, J., Peng, P., & Luo, L. (2020). The Relation Between Family Socioeconomic Status and Academic Achievement in China: A Meta-analysis. *Educational Psychology Review*, *32*(1), 49-76. https://doi.org/10.1007/s10648-019-09494-0

Price, C. B., & Price-Mohr, R. (2021). Exploring gender differences in primary school computer programming classes: A study in an English state-funded urban school. *Education 3-13*, 1-14. https://doi.org/10.1080/03004279.2021.1971274

Resnick, M. (2014). *GIVE P'S A CHANCE: PROJECTS,*

PEERS, PASSION, PLAY. 8.

Resnick, M., & Silverman, B. (2005). Some reflections on designing construction kits for kids. *Proceeding of the 2005 Conference on Interaction Design and Children - IDC '05*, 117-122. https://doi.org/10.1145/1109540.1109556

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology*, *111*(5), 764-792. https://doi.org/10.1037/edu0000314

DOI: 10.34641/ctestem.2022.477

Log-Based Multidimensional Measurement of CT Acquisition

Rotem ISRAEL-FISHELSON, Arnon HERSHKOVITZ*
School of Education, Tel Aviv University
rotemisrael@tauex.tau.ac.il, arnonhe@tauex.tau.ac.il

ABSTRACT
Computational thinking (CT) has been proven challenging to conceptualize and assess. When assessing CT using problem-solving tasks, it is commonly measured based on achievements, that is, in a unidimensional summative way. However, this traditional measurement neglects to consider vital components of the learning progress, which may produce a richer, formative assessment. Using the log files drawn from an online learning platform for CT (Kodetu), we suggest a nuanced evaluation of CT acquisition which consists of four variables: number of attempts to solve a problem; time to solution; application of newly presented CT concept; and solution originality. The research population included 189 middle-school students who participated in a workshop aimed at promoting CT and creativity. Using a learning analytics approach, we analyzed data from a log file documenting 1478 student-task pairs. Findings suggest that these variables share some common features that make them suitable for assessing CT acquisition. Furthermore, the variables grasp different aspects of the learning progress; hence, taken together, they allow for a richer evaluation of CT acquisition. These results shed light on the importance of using diverse metrics to examine CT and contribute to the proliferation of assessment practices.

KEYWORDS
Computational thinking, assessment, CT concepts, achievements, originality

1. INTRODUCTION
A growing trend in educational systems looks to train students in vital skills such as problem-solving and computational thinking (CT). However, several aspects of CT make it challenging to quantify and evaluate it reliably (Blikstein, 2011). The many operational definitions available in the literature and the lack of consensus regarding CT's core features and competencies make it challenging to establish a uniform assessment approach (Cutumisu et al., 2019; Grover et al., 2015). Moreover, because CT is a relatively ill-defined and complex construct, various methods may focus on different dimensions of CT (Weintrop et al., 2021). Various assessment methods were developed and studied following the proliferation of CT-related initiatives. Román-González et al. (2019) proposed a valuable classification of assessment tools based on their evaluation approach. Tang et al. (2020), who reviewed 96 journal articles, offered a more concise classification. Four categories emerged from their analysis: traditional assessment composed of selected- or constructed-response questions, portfolio assessment, survey, and interview. Traditional tests are the most common evaluation method, and they mainly examine the correctness of items for summative purposes (Cutumisu et al., 2019; Metcalf et al., 2021). Such tests, along with surveys, interviews, or observations, are authentic and can lead to a deep understanding of the learning outcomes and required skills (Guenaga et al., 2021). However, focused on the scores achieved, they cannot capture the learning process and draw insights from it (Fields et al., 2019).

Portfolio assessment is used to evaluate CT skills mainly through projects and artifacts, using different rubrics for grading the level of achievement and understanding (Metcalf et al., 2021). Dr. Scratch, for example, draws insights on the application of CT concepts through the analysis of the coding blocks used (Moreno-León et al., 2015). Since CT is not a binary state which developed over time, it is crucial to analyze students' trajectories along the learning experience (Brennan & Resnick, 2012). Portfolio assessment supports such exploration. It enables the capture of the program iterations and identifies patterns and difficulties while solving various challenges (Metcalf et al., 2021). However, this method is often based on human ranking and performed manually (Tang et al., 2020).

Data mining and learning analytics methods are focused on the learning process and are based on automatic analysis of the learning platforms' logged data. They are practical approaches for predicting students' success (Emerson et al., 2019) and detecting difficulties while acquiring CT concepts over time (Román-González et al., 2019). In addition, such methods can help evaluate knowledge acquisition by aggregating students' achievements in learning tasks (Kong, 2019).

Moreover, different indicators that emerge from the logged data can be used to analyze CT's development and provide a multidimensional perspective of CT. Examples of such indicators used in recent studies are the duration and number of attempts to reach a solution, the length of the code, and the number of changes in the code (Eguíluz et al., 2017; Guenaga et al., 2021). Indeed, different indicators, such as score, completion rate, and completion time, provide additional layers of information. However, most of the studies have used such measures in a unidimensional manner and referred to them all as measuring the acquisition of CT. As was recently shown in another domain, this is not necessarily the case (Haleva et al., 2021).

Therefore, this study investigates the processes of CT concept acquisition in a game-based learning platform by performing a multidimensional analysis. We take a learning analytics approach to study the associations between four variables: number of attempts to solve a

©Author(s) 2022. This work is distributed under the Creative Commons Attribution 4.0 License. This license allows anyone to redistribute, mix and adapt, as long as credit is given to the authors.

problem; time to solution; application of newly presented CT concept; and solution originality.

2. METHODOLOGY

2.1. The Learning Environment: Kodetu

Kodetu is a block-based online platform for acquiring CT. The platform is aimed primarily at elementary and middle school students with or without previous coding experience. It offers predefined challenges and enables the independent creation of challenges for the benefit of research or learning. Each challenge is comprised of levels in which the user has to route an astronaut on a given path to a marked destination by dragging coding blocks available in the workspace. Moving to the next level is possible only upon completing the current level, i.e., bringing the astronaut to the marked destination. Notably, users can repeat a level upon completing it and submit another solution. The platform logs all the actions performed by the user. See Guenaga et al. (2021) for further details on this platform.

For this research, we created a dedicated challenge comprised of eight levels that deal with three CT concepts: *Sequences, Loops,* and *Conditionals*. Levels 1-3 focused on the concept of *Sequences*. These levels include blocks representing instructions to move forward and turn right or left. Levels 4-6 present the concept of *Loops*. These levels have a “While” loop that repeats the operations until the astronaut's destination is reached. Finally, levels 7-8 present the concept of *Conditionals* (see Figure 1, for example). In these levels, the users are first presented with a block representing an "if" condition and then with a block representing an "if-else" condition. Each level is built on the previous concept presented.

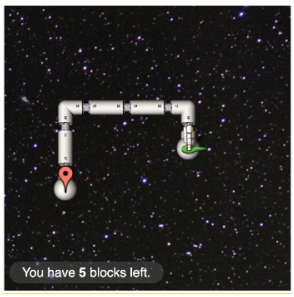


Figure 1. Example of Level 7

2.2. Population and Dataset

The sample comprised 189 ninth-grade students, 15-14 years old. Of them, 40% boys and 60% girls. The vast majority of the students (87%) had no prior coding experience, and 59% had a high affinity for technology. Students were given 80 minutes to solve eight dedicated levels created within the Kodetu platform. This was their first experience with Kodetu, as part of a broader study to examine the associations between CT and creativity (Israel-Fishelson & Hershkovitz, 2022). Data were collected anonymously, with a unique ID assigned to each student. The data were analyzed from the log files retrieved from the Kodetu platform. The log file included 21,784 rows, each representing an action taken by a student, including the users' unique ID, the level at which it was taken, the solution provided (both in Java code and the blocks used), its result [Success, Failure, Timeout, Error], and its timestamp. All statistical analyses were conducted using JASP version 0.16.1.

2.3. Research Variables

2.3.1. Computational Thinking

Three variables were used to measure the acquisition of CT, each computed first for each level separately and then averaged across all levels:

- *Solution Attempts* [#] – counting all solution attempts, including correct and incorrect ones (M=4.82, SD=2.27).
- *Completion Time* [min.] – calculated as the difference between the time of loading a level and the time of moving on to the next level (M=2.35, SD=1.52).
- *Concept Utilization* [0/1] – calculated by checking whether the concept-related blocks were used in the submitted solution. In levels 1-3, all the blocks were related to *sequences*, i.e., moving forward and turning right or left. In levels 4-5, it is examined whether *Loops* have been applied by extracting the command "FOREVER" from the code. Finally, in levels 6-8, it is examined whether *Conditionals* have been used by extracting the command "IF" from the code.

2.3.2. Computational Creativity

To measure the expression of creativity within the Kodetu platform, we have calculated *Solution Originality* as reflected by the frequency of a particular solution among all correct solutions, assessed on a scale of 0-1. In cases when an individual participant submitted several correct solutions, the average frequency of the solutions was taken. This measure was calculated for each level separately and then averaged across all levels (M=0.49, SD=0.1).

3. FINDINGS

To understand how the four research variables are associated with each other, we first checked their values along the game based on CT concepts. Then, we tested for correlations between pairs of them. Finally, we used cluster analysis to classify the participants into groups based on these variables.

3.1. Values of the Research Variables Along the Game

To better understand the acquisition of the three concepts, i.e., *Sequences, Loops,* and *Conditionals,* we conducted 12 pair-wise t-tests between each of the three concepts for each of the four research variables.

For the Solution Attempt variable, we have found an increase in the number of attempts submitted as the challenge progressed (see Figure 2), indicating that the concept of *Sequences* (levels 1-3) required the fewest attempts, followed by *Loops* (levels 5-6) and *Conditionals* (levels 7-8). These differences were significant (at $p<0.001$) with a medium-high effect size (*Sequences-Loops*: d=-0.45; *Loops-Conditionals*: d=-0.96; *Sequences-Conditionals*: d=-1.16). Note that the increase in Solution Attempts variables is not linear.

A similar trend was found for the *Completion Time* variable, as reflected by the increase in the average time to complete the levels along the challenge (see Figure 3). The time to complete the levels dealing with *Sequences* was significantly the shortest, followed by *Loops* and *Conditionals*. As in the case of *Solution Attempts*, this increase in values is also not linear.

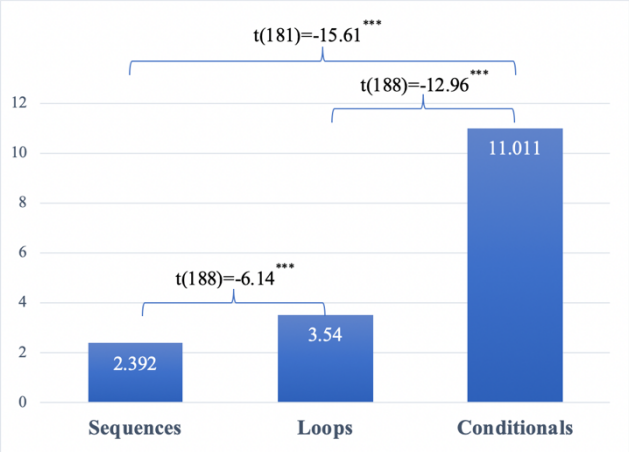


Figure 2. Comparing *Solution Attempts* by CT Concepts (*** p<0.001)

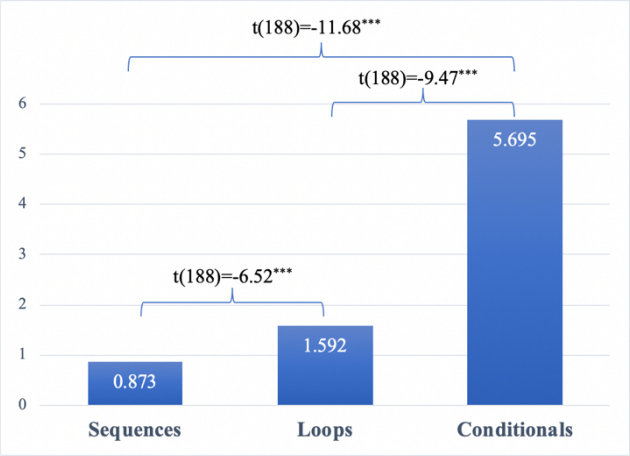


Figure 3. Comparing *Completion Time* by CT concepts (*** p<0.001)

As for *Concept Utilization*, we found that in levels related to the concept of *Loops*, there was a high usage rate of the designated *Loops* block (a "Do-While" loop) in the code. This rate was significantly higher compared to the usage of the designated *Conditionals* blocks ("IF" and "IF-ELSE" blocks) in the related levels (see Figure 4). Note that it is impossible to complete the levels dealing with *Sequences* without using the sequence-related blocks (moving forward and turning right or left), so all the solutions for these levels have implemented the concept of *Sequences*. Therefore, for testing for differences between *Concept Utilization* means in *Loops* and *Conditionals* with *Sequences*, we used a one-sample t-test, comparing them to 1; both were significantly lower than that value.

As for *Solution Originality*, we found that students provided more original solutions as the challenge progressed. The solutions for levels dealing with the concept of *Sequences* were found to be the least original ones, followed by *Loops* and *Conditionals*. These findings were significant (at p<0.001) with a medium-high effect size (*Sequences-Loops*: d=-0.57; *Loops-Conditionals*: d=-0.9; *Sequences-Conditionals*: d=-1.31), and depict a linear increase (see Figure 5).

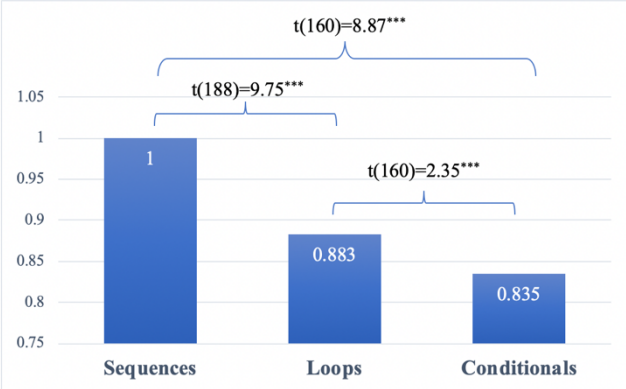


Figure 4 Comparing *Concept Utilization* by CT concepts (*** p<0.001)

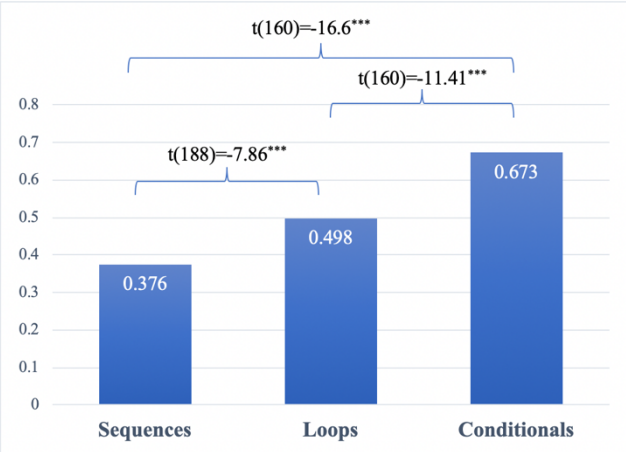


Figure 5. Comparing *Solution Originality* by CT concepts (*** p<0.001)

3.2. Correlations Between the Research Variables

Next, we examined the correlation between the four research variables for the concepts of *Loops* and *Conditional*. Observing these correlations points to three patterns (see Table 1). First, *Solution Attempts*, *Completion Time,* and *Solution Originality* were all positively correlated with each other. The more solutions students submitted, the more time it took them, and the more original their solutions were. For *Solution Attempts* and *Completion Time* specifically, we see very high coefficient values in *Loops* and *Conditionals* (0.85 and 0.69, respectively). This shows the strong connection between these two measures.

A positive correlation was also found between *Concept Utilization* and *Solution Originality* in the levels related to the concept of *Conditionals*. The more students applied the concept of *Conditionals*, the more original their solutions in these levels were. However, it is important to note that the coefficient value, in this case, was low, indicating a low connection between them.

In contrast, a significant negative correlation was found between *Concept Utilization* and *Solution Originality* in the levels related to *Loops*. The more students applied the concept of *Loops*, the less original their solutions were. Also, in this case, the coefficient value was low, indicating a low connection between the measures.

Notably, there were no significant correlations between *Concept Utilization* and *Solution Attempts*, and between *Concept Utilization* and *Completion Time*, neither for *Loops* nor for *Conditions*.

Table 1. Correlations between Research Variables, Per CT Concept

	Var 1	Var 2	ρ
Loops	Solution Attempts	Completion Time	0.85***
	Solution Attempts	Concept Utilization	-0.05
	Solution Attempts	Solution Originality	0.26***
	Completion Time	Concept Utilization	0.01
	Completion Time	Solution Originality	0.21**
	Concept Utilization	Solution Originality	-0.17*
Conditionals	Solution Attempts	Completion Time	0.69***
	Solution Attempts	Concept Utilization	0
	Solution Attempts	Solution Originality	0.26***
	Completion Time	Concept Utilization	0.04
	Completion Time	Solution Originality	0.22**
	Concept Utilization	Solution Originality	0.21**

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

3.3. Clustering Students by the Research Variables

After examining the different variables and their behavior, we analyzed the research population on a higher granularity level. To that end, we used an unsupervised hierarchical clustering algorithm based on the four research variables.

The hierarchical clustering algorithm aims to partition and group objects based on their similarities. The similarity between the cluster was measured using Pearson's distance, using Ward.D linkage (Ward, 1963), with variables scaled by a Z-score standardization of a mean of 0 and a standard deviation of 1. The elbow method indicated that five is the optimal number of clusters for our dataset. These clusters represent five sub-populations with distinct characteristics, as detailed below (see Figure 6 and Table 2).

Cluster 1 (N=41) includes students who quickly solved the challenge with the least number of attempts. Their application of the CT concepts was among the highest, but their solutions were the least original. Cluster 2 (N=37) includes students who required fewer attempts to solve the levels and the least time for completion. Additionally, they provided the most original solution, and their utilization of the concepts was relatively high. The students in this cluster had the best performance. Students in Cluster 3 (N=44) demonstrated mediocre performance. They were able to solve the levels in the shortest time with a low number of attempts. They were relatively original in providing the solutions but did little use of the concepts learned. Cluster 4 (N=39) included low-performing students. They solved the levels in the longest time and with the most attempts. They provided the second to lowest original solutions, and their implementation of the concepts was also low. Cluster 5 (N=28) also included students with moderate performance. They submitted many solutions but were able to solve the challenge in a short time. Their usage of the concepts was the highest, but their solutions were not so original.

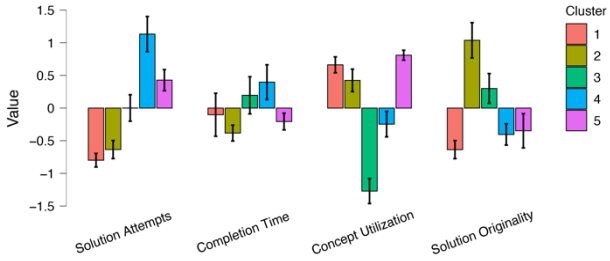


Figure 4. Clusters representing students' behavior according to the research variables

Table 2. Cluster Means. Grey background marks the value in each column that is indicative of the highest performance in CT acquirement or originality; numbers in italics indicate, for each column, the lowest performance in CT acquirement or originality

Cluster	N	Solution Attempts	Completion Time	Concept Utilization	Solution Originality
1	41	-0.798	-0.102	0.662	-0.636
2	37	-0.635	-0.383	0.424	1.038
3	44	0.001	0.196	*-1.271*	0.3
4	39	*1.133*	*0.397*	-0.246	-0.404
5	28	0.428	-0.206	0.81	-0.348

4. DISCUSSION

CT is most often measured by achievements in a unidimensional summative way. However, such an evaluation approach neglects to consider essential factors of the learning process that may produce a richer assessment. This study investigated a multidimensional evaluation of CT acquisition by 189 middle-school students who used an online gamed-based platform. We evaluated students' performance according to four dimensions: *Solution Attempts* (number of attempts to solve a problem); *Completion Time* (time to complete a challenge); *Concept Utilization* (application of newly presented CT concept); and *Solution Originality* (frequency of a correct solution in the set of all correct solutions). Our findings indicate complex relationships

between these measures and suggest that they may capture different aspects of the learning process.

Superficially, as it seems from the exploratory analysis, the four variables demonstrate a similar learning behavior. *Solutions Attempts* and *Completion Time* increased as the game progressed, hence may be seen as proxies for difficulty. Moreover, both variables increase in a non-linear way. *Concept Utilization* decreased as the game progressed, demonstrating that correct solutions were often not implementing newly-taught concepts but rather relied on previous knowledge – which, again, reflects the increased difficulty. Finally, *Solution Originality* increased, which may be explained by the fact that the overall set of solutions within our research population increased along the game, echoing the behavior depicted by *Concept Utilization*. Indeed, both variables decrease or increase relatively linearly. This is in line with previous studies, which pointed out some difficulties and misconceptions regarding the concepts of *Loop*s and *Conditionals* (Grover & Basu, 2017; Israel-Fishelson & Hershkovitz, 2019; Weintrop & Wilensky, 2015). Sleeman et al. (1986) argued that such difficulties and misconceptions could stem from a limited understanding of the execution of "if" and "if-else" conditions, as well as from having a faulty understanding of which lines of codes would repeat themselves in "for" and "while" structures and the number of times the code would run.

However, further analysis has shown that the picture is more complex than this. First, correlations between pairs of variables slightly change when tested in different game levels. *For example, concept Utilization* and *Solution Originality* were negatively associated while engaging with the *Loops*-related levels and positively associated with the *Conditionals*-related levels. Additionally, *Solution Attempts* and *Completion Time* were more strongly correlated in *Loops*- than in *Conditionals*-related tasks (in both cases, the correlation was positive).

Second, when clustering students based on their behavior throughout the game, we observe even more complicated relationships. For example, we have a cluster where *Solution Attempts* and *Completion Time* are both low, on average (Cluster 2), a cluster where they are both relatively high (Cluster 4), and a cluster when one of them is high, and the other is low (Cluster 5). Similarly, for *Concept Utilization* and *Solution Originality*, we have clusters that demonstrate different behaviors (respectively): high-high (Cluster 2), high-low (Cluster 3), low-high (Clusters 1, 5), low-low (Cluster 4). Taken together, these findings suggest that CT acquirement measures depend on both personal and contextual characteristics. Indeed, previous studies have shown the importance of contextual factors in the acquisition of CT, and creativity, which is seemingly associated with personal characteristics, is also impacted by contextual factors (Hershkovitz et al., 2019; Israel-Fishelson & Hershkovitz, 2021).

As clearly evident by the cluster analysis, the misalignment between the different measures strengthens the notion that they may each grasp a different aspect of learning. Most easily explained is *Completion Time*, that is, time on task. This measure was shown in other contexts and settings to be impacted by factors other than "knowing" the subject matter, e.g., skill level or graphical user interface, hence it is not necessarily correlated with other, more traditional, measures like achievements or the number of attempts to solve a problem (Goldhammer et al., 2014; Haleva et al., 2021; Hershkovitz et al., 2019).

Our creativity measure, *Solution Originality*, is overall associated with *Solution Attempts* and *Concept Utilization.* This may be explained by the positive association previously suggested between creativity and difficulty (Espedido & Searle, 2018). It also echoes Epstein et al.'s (2008) claim that people can increase their production of new ideas and creative expression when facing challenging problem-solving situations. However, the associations between our creativity measure and our *Concept Utilization* were alternately positive and negative (when tested for each topic separately). This reflects that creativity and knowledge are not necessarily tied together (Edmonds & Candy, 2002). It is possible that students who had difficulty in solving the levels adopted a tinkering strategy which was found effective when learning to program (Berland et al., 2013). Thus, for *Loops* levels, which are considered easier, lower originality rates were observed compared to the *Conditionals* levels, which are considered harder.

This study contributes to the growing body of knowledge on CT assessment while emphasizing the importance of using diverse metrics to examine CT. Taking a log-based approach, we were able to identify nuanced relationships between the different measures throughout the learning process. These associations should be further investigated, on a larger scale, in other populations and contexts. Still, we hope that these findings will encourage researchers to consider the combination of different CT assessment indices to get a more fine-grained, rich assessment.

5. REFERENCES

Baker, R. S. J. d. (2007). Is gaming the system state-or-trait? Educational data mining through the multi-contextual application of a validated behavioral model. *Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling*, 76–80. http://www.columbia.edu/~rsb2162/B2007B.pdf

Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564–599.

Blikstein, P. (2011). Using learning analytics to assess students' behavior in open-ended programming tasks. In P. Long, G. Siemens, G. Conole, & D. Gasevic (Eds.), *Proceedings of the 1st International Conference on Learning Analytics and Knowledge - LAK '11* (pp. 110–116). ACM Press.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In C. A. Tyson & A. F. Ball (Eds.), *Proceedings of the 2012 Annual Meeting of the American Educational Research Association* (pp. 1–25). American Educational Research Association.

Cutumisu, M., Adams, C., & Lu, C. (2019). A scoping review of empirical research on recent computational thinking

assessments. *Journal of Science Education and Technology*, *28*(6), 651–676. https://doi.org/10.1007/s10956-019-09799-3

Eguíluz, A., Guenaga, M., Garaizar, P., & Olivares-Rodríguez, C. (2017). Exploring the progression of early programmers in a set of computational thinking challenges via clickstream analysis. *IEEE Transactions on Emerging Topics in Computing*, *8*(1), 256–261. https://doi.org/10.1109/TETC.2017.2768550

Emerson, A., Smith, A., Smith, C., Rodríguez, F., Wiebe, E., Mott, B., Boyer, K., & Lester, J. (2019). Predicting early and often: Predictive student modeling for block-based programming environments. In C. F. Lynch, A. Merceron, M. Desmarais, & R. Nkambou (Eds.), *Proceedings of the 12th International Conference on Educational Data Mining* (pp. 39–48).

Epstein, R., Schmidt, S. M., & Warfel, R. (2008). Measuring and training creativity competencies: Validation of a new test. *Creativity Research Journal*, *20*(1), 7–12. https://doi.org/10.1080/10400410701839876

Espedido, A., & Searle, B. J. (2018). Goal difficulty and creative performance: The mediating role of stress appraisal. In *Human Performance* (Vol. 31, Issue 3, pp. 179–196). https://doi.org/10.1080/08959285.2018.1499024

Fields, D. A., Lui, D., & Kafai, Y. B. (2019). Teaching computational thinking with electronic textiles: Modeling iterative practices and supporting personal projects in exploring computer science. In *Computational Thinking Education* (pp. 279–294). Springer Singapore. https://doi.org/10.1007/978-981-13-6528-7_16

Goldhammer, F., Naumann, J., Stelter, A., Tóth, K., Rölke, H., & Klieme, E. (2014). The time on task effect in reading and problem solving is moderated by task difficulty and skill: Insights from a computer-based large-scale assessment. *Journal of Educational Psychology*, *106*(3), 608–626. https://doi.org/10.1037/a0034716

Grover, S., & Basu, S. (2017). Measuring student learning in introductory block-based programming: examining misconceptions of loops, variables, and boolean logic. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE '17*, 267–272.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, *25*(2), 199–237.

Guenaga, M., Eguíluz, A., Garaizar, P., & Gibaja, J. (2021). How do students develop computational thinking? Assessing early programmers in a maze-based online game. *Computer Science Education*, *31*(2), 259–289. https://doi.org/10.1080/08993408.2021.1903248

Haleva, L., Hershkovitz, A., & Tabach, M. (2021). Students' activity in an online learning environment for mathematics: The role of thinking levels. *Journal of Educational Computing Research*, *59*(4), 686–712. https://doi.org/10.1177/0735633120972057

Hershkovitz, A., Sitman, R., Israel-Fishelson, R., Eguíluz, A., Garaizar, P., & Guenaga, M. (2019). Creativity in the acquisition of computational thinking. *Interactive Learning Environments*, *27*(5–6), 628–644. https://doi.org/10.1080/10494820.2019.1610451

Israel-Fishelson, R., & Hershkovitz, A. (2019). Persistence and achievement in acquiring computational thinking concepts: A large-scale log-based analysis. In S. Carliner (Ed.), *Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*. Association for the Advancement of Computing in Education (AACE).

Israel-Fishelson, R., & Hershkovitz, A. (2021). Micro-persistence and difficulty in a game-based learning environment for computational thinking acquisition. *Journal of Computer Assisted Learning*, *37*(3), 839–850. https://doi.org/10.1111/jcal.12527

Israel-Fishelson, R., & Hershkovitz, A. (2022). One small step to man, a giant step to computational thinking: Improving student performances by promoting creativity. Seventeenth Chais Conference for the Study of Innovation and Learning Technologies, 36–46 [Hebrew].

Kong, S. (2019). Components and methods of evaluating computational thinking for fostering creative problem-solvers in senior primary school education. In S. Kong & H. Abelson (Eds.), *Computational Thinking Education* (pp. 119–142). Springer.

Metcalf, S. J., Reilly, J. M., Jeon, S., Wang, A., Pyers, A., Brennan, K., & Dede, C. (2021). Assessing computational thinking through the lenses of functionality and computational fluency. *Computer Science Education*, *31*(2), 199–223. https://doi.org/10.1080/08993408.2020.1866932

Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. *RED. Revista de Educación a Distancia*, *15*(46), 1–23. https://doi.org/10.6018/red/46/10

Román-González, M., Moreno-León, J., & Robles, G. (2019). Combining assessment tools for a comprehensive evaluation of computational thinking interventions. In S. Kong & H. Abelson (Eds.), *Computational Thinking Education* (pp. 79–98). Springer.

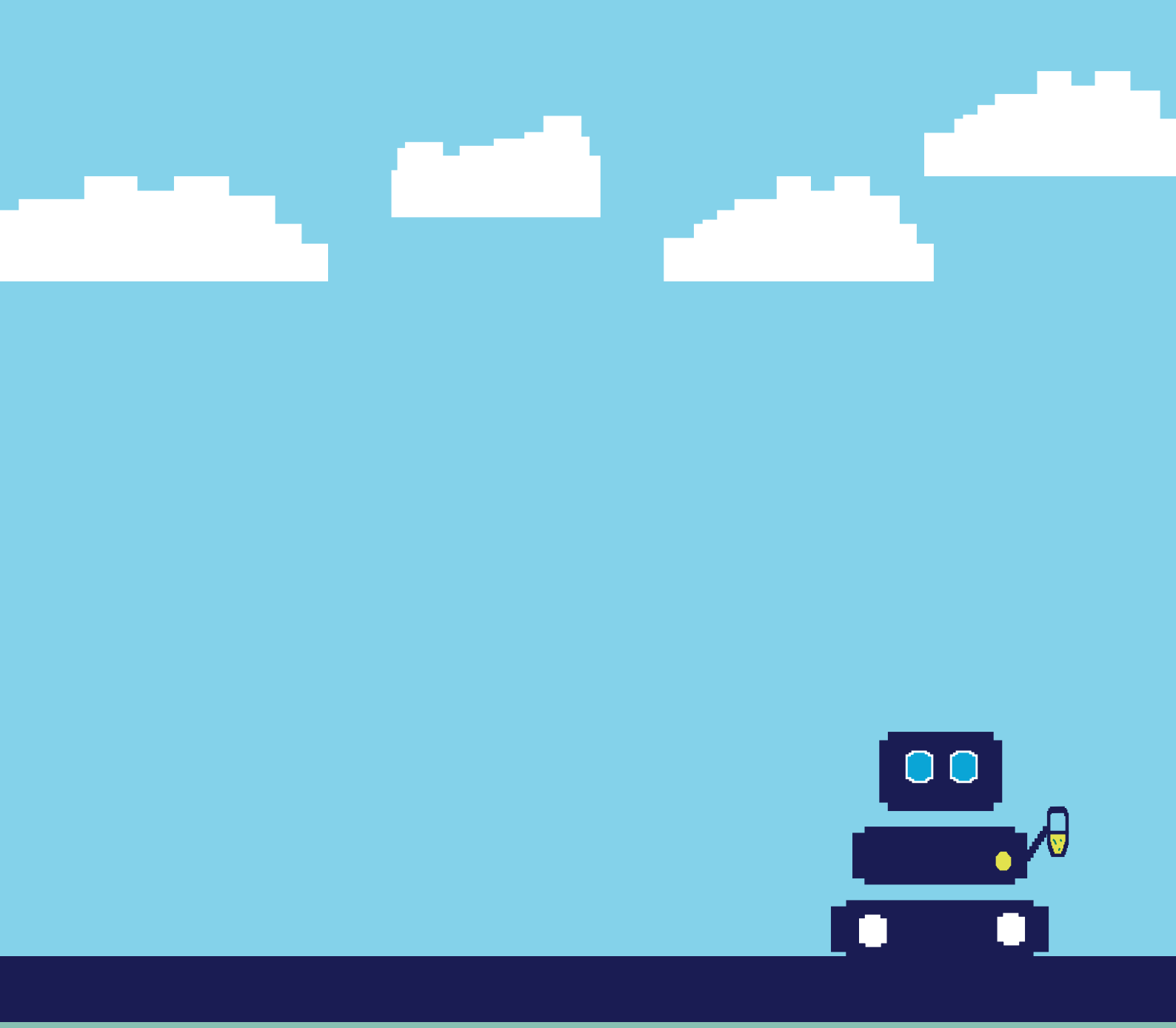
Sleeman, D., Putnam, R. T., Baxter, J., & Kuspa, L. (1986). Pascal and high school students: A study of errors. *Journal of Educational Computing Research*, *2*(1), 5–23.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, *148*. https://doi.org/https://doi.org/10.1016/j.compedu.2019.103798

Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*(301), 236–244.

Weintrop, D., & Wilensky, U. (2015). Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. *ICER*, *15*, 101–110.

Weintrop, D., Wise Rutstein, D., Bienkowski, M., & McGee, S. (2021). Assessing computational thinking: an overview of the field. *Computer Science Education*, *31*(2), 113–116. https://doi.org/10.1080/08993408.2021.191838



TU Delft OPEN

@2022 TU Delft OPEN Publishing
ISBN: 978-94-6366-563-6
ISSN: 2664-5661
DOI: https://doi.org/10.34641/mg.37

9 789463 665636 >