Computational Thinking Assessment in Higher Education

An Exploration of Computational Thinking Assessment and Integration in Higher Education in the Netherlands

Zhang, X.

DOI
10.4233/uuid:62afaacb-ba31-4fc9-99ca-e8bcf75ce46b

Publication date
2025

Document Version
Final published version

Citation (APA)
Zhang, X. (2025). *Computational Thinking Assessment in Higher Education: An Exploration of Computational Thinking Assessment and Integration in Higher Education in the Netherlands*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:62afaacb-ba31-4fc9-99ca-e8bcf75ce46b

COMPUTATIONAL THINKING ASSESSMENT IN HIGHER EDUCATION

AN EXPLORATION OF COMPUTATIONAL THINKING ASSESSMENT AND INTEGRATION IN HIGHER EDUCATION IN THE NETHERLANDS

XIAOLING ZHANG

Computational Thinking Assessment in Higher Education

An Exploration of Computational Thinking Assessment and Integration in Higher Education in the Netherlands

Computational Thinking Assessment in Higher Education

An Exploration of Computational Thinking Assessment and Integration in Higher Education in the Netherlands

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
Thursday 17th, April 2025 at 15:00 o'clock

by

Xiaoling ZHANG

Master of Science in Computer Science, Leiden University, the Netherlands,
born in Chongqing, China

This dissertation has been approved by the promoters.

Composition of the doctoral committee:

Rector Magnificus	chairperson
Prof. dr. M. M. Specht	Delft University of Technology, *promotor*
Prof. dr. ir. F. F. J. Hermans	VU Amsterdam, *promotor*
Dr. ir. E. Aivaloglou	Delft University of Technology, *copromotor*

Independent members:

Prof. dr. M. M. Bonsangue	Leiden University
Prof. dr. A. van Deursen	Delft University of Technology
Dr. E. Rahimi	Open University of the Netherlands
Prof. dr. -Ing. U. Schroeder	RWTH Aachen University, Germany
Prof. dr. ir. G. J. P. M. Houben	Delft University of Technology, reserve member

Keywords: Computational Thinking, Assessment, Integration, Higher Education, the Netherlands

Printed by: Gildeprint

Cover by: Xiaoling Zhang (Pictures from freepik were adopted)

ISBN 978-94-6496-362-5

An electronic copy of this dissertation is available at
https://repository.tudelft.nl/.

EDUCATION IS THE KINDLING OF A FLAME, NOT THE FILLING OF A VESSEL.

SOCRATES

Contents

1

Introduction

Computational thinking is not only something programmers must know, but it is also a thinking tool for understanding our technology-infused social world.

Peter J. Denning & Matti Tedre, 2019

1

1.1. Computational Thinking Assessment in Higher Education

Over the past century, technological advancements have increased our reliance on computers, systems, data, and automation, transforming many aspects of life. While smartphones, tablets, computers, and other connected devices offer numerous benefits, they also introduce new challenges. These advances require new skills and knowledge, yet not all students are adequately prepared. This situation demands education systems to adapt and prepare students for future environments.

How can we thrive in an increasingly digital era? What should we teach to empower students to navigate their path in different scenarios? These questions prompt further exploration especially also in the field of programming education: Is coding essential for everyone, and is proficiency in coding alone sufficient for success? Many find the concept of Computational Thinking (CT) pivotal. CT, introduced by Computer Science professor Jeanette Wing in her influential 2006 article [1], offers a perspective rooted in Computer Science tools and methods. Since then, substantial attention and funding various organizations and leading scholars have fueled research into defining, assessing, and teaching CT [2, 3]. Despite ongoing debates about its precise definition, efforts to integrate CT into educational programs are progressing globally.

However, there is little guidance on incorporating CT into higher education curricula as research often focuses on K-12 education. Experts have not reached a consensus on how CT should be taught, by whom, and how students, faculty, and staff should be held accountable for acquiring CT skills, knowledge, and attitudes [4]. In addition to the lack of research on CT in Higher Education, the nature of curriculum development in this setting poses extra challenges for large-scale improvements. Unlike K-12 education, Higher Education grants professors and faculty a high degree of autonomy in developing and managing their curricula. The high degree of autonomy leads to various implementations of CT integration in higher education and different practices assessing it.

In the Netherlands specifically, CT education has been promoted in various ways. According to Yadav et al. [5], the Academy's report from the Royal Dutch Academy of Arts and Sciences (an advisory body to the Dutch government) shaped an active discussion in 2013. Moreover, there were explorative studies on CT integration in Dutch Education, for example by Thijs et al.[6]. However, no official national policy was published on integrating CT in the curriculum. Meanwhile, although scholars have identified the need for more implementations of CT within higher education [7, 8], most existing studies on CT in the Netherlands have focused on the K-12 context [9].

Assessment plays a crucial role in CT education by providing valuable insights into student learning and progress. For example, it helps educators gauge the effectiveness of teaching methods, identify areas where students may need additional support, and measure overall achievement of learning objectives. Moreover, effective assessment strategies also inform instructional decisions, allowing educators to tailor their teaching approaches to better meet the needs of their students.

Furthermore, assessment fosters accountability and transparency in education by providing stakeholders with evidence of educational outcomes. In recent years, the Dutch government has placed increasing emphasis on assessment quality in higher education [10]. According to Standard 2 of the Dutch Qualification Framework [11], the quality of the teaching and learning environment should be designed to help students achieve the intended learning outcomes of the program curriculum.

Though given the importance of assessment in CT education, there is comparatively less emphasis on how to reliably and validly assess students' CT skills in diverse academic contexts within higher education [12, 13]. Due to different understandings of CT, various necessity on specific CT dimensions for different domains, and the fact that teachers are with high autonomy for operationalising CT in their practices, we can find various CT assessments. Several approaches to assessing CT have been tried, including performance-based assessments, multiple-choice tests, and rubric-based evaluations of student projects [7, 13, 14].

While these methods offer some insights, they often fail to fully capture the depth and breadth of CT as a cognitive skill set. For instance, performance-based assessments can be too narrowly focused on specific tasks, missing broader CT skills like abstraction and generalization. Multiple-choice tests may fail to assess higher-order thinking and the ability to apply CT in novel contexts. Rubric-based evaluations, while more flexible, often lack the rigor needed for consistent and objective assessment across different instructors and institutions.

Current practices often focus on assessing CT in programming contexts in various domains. Moreover, CT assessments often do not capture the need for different levels of CT skills [15]. In contrary, some assessment approaches may increase cognitive load, particularly if they require students to simultaneously learn new tools or languages while demonstrating their CT skills. Additionally, CT assessments are often tailored to specific contexts or disciplines, making it difficult to generalize findings or apply assessment tools across different educational settings. Last but not least, there is no consensus on a standardized framework for assessing CT, leading to inconsistencies in how CT skills are measured and interpreted.

Given the identified gap in the literature and the shortcomings of existing assessment methods, there is a compelling need to develop more comprehensive, standardized, and cognitively grounded approaches to assessing CT in higher education. This thesis aims to address this need by exploring innovative assessment strategies that can more accurately reflect the cognitive processes involved in CT and be applied across a range of disciplines and educational contexts. To achieve this, in this thesis, theoretical and practical knowledge has been analysed by reviewing the literature, surveys have been deployed on the integration of CT into teaching practices, and assessment has been designed for measuring CT.

1.2. OBJECTIVES

Based on the background provided in Chapter 1.1, we idenfity the following overarching research questions and their corresponding research objectives for this thesis:

1

- **Research Question 1** What is the state of the art of Computational Thinking assessment in higher eduaction?
- **Objective 1** Determine the investigated dimensions of CT assessment in higher education in the literature.
- **Research Question 2** What is the current understanding of the practical integration of CT in higher education?
- **Objective 2** Investigate CT integration in higher education in practice in STEM courses and curricula.
- **Research Question 3** How can an effective assessment instrument be designed to measure CT skills based on current practices and assessment strategies in higher education?
- **Objective 3** Based upon investigated aspects of CT assessment and practical information from CT integration in higher education, design an assessment instrument for measuring CT skills.

1.3. Overview of Research Methods

Theory and practice for educational innovation and research are deeply interconnected, each enriching and advancing the other. Accordingly, this thesis begins by investigating existing literature on CT assessment practices to develop a robust theoretical framework. Building on this foundation, we then explore current CT practices, applying insights from the literature and our analysis to design an assessment tool. This tool will test our theoretical hypotheses, allowing us to make meaningful contributions to the field by bridging the gap between theory and practice. Three chapters (Ch.2 to Ch.4) in this dissertation focus on gathering and analyzing existing literature and other resources on CT assessment and integration in higher education. The remaining two contributions (Ch.5 and Ch.6) focus on investigating CT integration in practice and the design of assessment for measuring students' CT skills.

To address **Objective 1** presented in Ch.1.2, we start with a systematic umbrella review on CT assessment in higher education in Ch.2, which provides a broad synthesis of existing reviews, offering a high-level understanding of the state of research on CT assessment practices. Following that, in Ch.3, we present a systematic review on CT assessment and delve deeper into individual studies to identify specific methodologies, tools, and outcomes, complementing the broader insights of the umbrella review.

To address **Objective 2** presented in Ch.1.2, we examine in Ch.4 how CT is currently embedded in higher education programs by analysing curricula from various STEM domains at Delft University of Technology to learn strengths and gaps in CT integration and assessment from practice. Moreover, in Ch.5, we present the results of a survey of teachers' understandings of CT and their intentions to integrate CT into their teaching practices provides crucial insights into educators' perspectives, which are key to successful implementation.

Regarding **Objective 3** presented in Ch.1.2, finally, in Ch.6, to address the practical need for instruments to measure CT skills (algortihmic abstraction specifically) for students from both CS and non-CS domains, we conduct an experimental lab study with 30 participants with a CT assessment tool designed according to findings from the preceding studies.

As presented above, we use various research methods to achieve the stated objectives. Table 1.1 presents an overview of the research methods, the type of work, and the information about participants (if applicable). The chapters involving human subjects, Chapter 5 and Chapter 6, were approved by the Ethical Review Board of the Delft University of Technology. These approvals ensure the privacy and safety of participants and proper data storage. Moreover, most chapters have elements of qualitative analysis, while others include both qualitative and quantitative analysis. Two studies involved 38 and 30 participants, respectively.

Together, these studies offer a holistic approach to improving CT assessment and integration in higher education.

Table 1.1.: Research methods used in the different studies

	Research method	**Qualitative Quantitative**	**Type of participant**	**Number of participants**
Chapter 2	Literature review	Qualitative	/	/
Chapter 3	Literature review	Qualitative	/	/
Chapter 4	Course description analysis	Qualitative	/	/
Chapter 5	Survey	Qualitative and Quantitative	Teachers in higher education	38
Chapter 6	Experimental Lab Study	Qualitative and Quantitative	Students in higher education	30

1.4. OUTLINE & CONTRIBUTIONS

In this dissertation, I explore CT assessment in literature and practice. The main contributions of this dissertation can be found in Chapters 2 - 6.

Chapter 2 presents an umbrella review of CT assessment in higher education. We examined 11 reviews to analyse the CT definitions, and aspects of assessment design, including a) methods, b) constructs, and c) contexts in which these have been applied. Most reviews encompass but are not limited to CT assessment in higher education. We identified 120 unique constructs and about 10 assessment methods. Although combining different assessment methods is recommended, we found no established guidelines for effective assessment design.

Chapter 3 provides a systematic review of 47 empirical studies on CT assessment in higher education, published between 1986 and 2021. These studies, primarily in STEM fields at the undergraduate level, often used summative assessments. Various CT constructs were assessed using both existing and newly developed tools in the studies.

Chapter 4 investigates the integration of CT into Master's level engineering curricula at Delft University of Technology. We analyzed the extent and manner of CT integration, along with the educational and assessment methods used. The results indicate that CT is largely integrated through lectures and programming assignments. Moreover, we found that understanding the problem's context, recognizing patterns in the problems, and organizing these patterns into solutions are key goals in many courses. These findings show potential directions for further CT integration.

Chapter 5 discusses a survey of 38 teachers, mainly in Engineering Education in the Netherlands, about their perceptions of CT and intentions to integrate it into teaching. Findings show that teachers often misunderstand the link between CT and Computer Science, have limited CT training, and vary in their views on integrating CT constructs. There is a strong positive correlation between teachers' performance expectancy, attitudes towards CT, and their intention to implement CT in their teaching.

Chapter 6 presents an assessment tool that used sorting algorithms as the focus to measure algorithmic abstraction skills (a key component of CT) for both CS and non-CS students. This tool was developed based on the PGK framework [16], which describes four levels of algorithmic abstraction: problem, algorithm, program, and execution levels. Our findings from this study indicate that both groups need abstraction skills at least at the *Algorithm Level*, with many requiring skills at the *Program Level* or the *Execution Level*. This is particularly crucial for STEM students, underscoring the necessity for tailored educational strategies for non-CS students, who exhibit a larger gap between their perceived and required mastery levels. Additionally, training and programming experience are strongly correlated with performance, highlighting the need for focused training in computational thinking, programming, and algorithmic abstraction.

Finally, in Chapter 7 we summarize our contributions. We do this by exploring the overarching themes for CT assessment in higher education in this dissertation. We also propose implications for practice and research directions for the future.

1.5. PUBLICATIONS

Publications in scientific conference proceedings and journals:

- **Zhang, X., Aivaloglou, F., and Specht, M. (2024).** A Systematic Umbrella Review on Computational Thinking Assessment in Higher Education. European Journal of STEM Education, 9(1), 02.

  This publication serves as core material for Chapter 2.

- **Zhang, X., Valle Torre, M., and Specht, M. (2022).** An Investigation on Integration of Computational Thinking into Engineering Curriculum at Delft University of Technology. In SEFI 2022 - 50th Annual Conference of the European Society for Engineering Education, Proceedings, 890–901. Universitat Politècnica de Catalunya. https://doi.org/10.5821

  This publication serves as core material for Chapter 4.

- **Zhang, X., Aivaloglou, F., and Specht, M. (2024).** Teachers' Intention to Integrate Computational Thinking Skills in Higher Education: A Survey Study in the Netherlands. 2024 IEEE Global Engineering Education Conference (EDUCON).

  This publication serves as core material for Chapter 5.

- **Zhang, X., Aivaloglou, F., and Specht, M.** Assessment of Algorithmic Abstraction Skills in Higher Education: An Application of the PGK Framework. Accepted by 2025 IEEE Global Engineering Education Conference (EDUCON).

  This publication serves as core material for Chapter 6.

Publications to which I contributed during my PhD but that are not included in the dissertation:

- **Zhang, X., Glahn, C., Fanchamps, L. J. A., Specht, M. M. (Eds.) (2022).** CTE-STEM 2022 | Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022. TU Delft. https://doi.org/10.34641/mg.37

- **Zhang, X., and Specht, M. (2022).** Towards a computer-assisted Computational Thinking (CT) assessment system in higher education. In: CEUR Workshop Proceedings, 3292, 12-21.

- **Zhang, X., and Specht, M. (2022).** A Review of Reviews on Computational Thinking Assessment in Higher Education. In Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022 (CTE-STEM), 98–103. https://doi.org/10.34641/CTESTEM.2022.472.

- **Cambaz, D. and Zhang, X. (2024).** Use of AI-driven Code Generation Models in Teaching and Learning Programming: a Systematic Literature Review. SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education. Association for Computing Machinery (ACM), p. 172-178 7 p. (SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education; vol. 1).

  I supervised this paper and this paper received a **Best Paper Award**.

Part I Theory and State of the Art on Computational Thinking Assessment

2

A Systematic Umbrella Review on Computational Thinking Assessment in Higher Education

Literature reviews play a critical role in scholarship because science remains, first and foremost, a cumulative endeavor.

vom Brocke et al., 2009

Parts of this chapter have been published in European Journal of STEM Education, 9(1), 02 (2024) [17]

Computational Thinking (CT) is considered a core 21st-century digital skill. Assessment is crucial and knowing what, who, when, how, and where to assess is important for assessment design. In this study, we conducted an umbrella review to gain insights regarding CT assessment in higher education. We analyzed 11 reviews, focusing on (1) the bibliographical and methodological characteristics of the reviews; (2) aspects relevant to assessment design, including a) assessed constructs, b) applied assessment methodologies, and c) assessment contexts. Our findings suggest an increased attention to this topic. Meanwhile, most reviews did not thoroughly examine existing reviews, and majority of the reviews focus on a wider scope than higher education. Moreover, We identified 120 unique assessed constructs and around 10 types of assessment methods. Though reviews suggested a combined use of distinct assessment methods, guidelines for appropriate assessment design are yet to be constructed. Based on the findings, we argue that it is necessary to explore different combinations of assessment design in various contexts to construct assessment guidelines.

2.1. INTRODUCTION

Computational Thinking (CT) is currently regarded as a core digital competency and it has been widely promoted in K-12 education as well as higher education settings. Wing [18] defined CT as a set of problem-solving skills with which people can formulate solutions for problems such that the solutions can be carried out by any information processing agent. More recently, CT has been stated as a thinking model that human beings utilize for problem-solving regardless of the rapid change of technology throughout history [19]. Existing studies investigated not only the definition of CT but also topics on various aspects of CT teaching and learning [4, 12, 20]. One can find a consensus on the importance of CT and various organizations have stressed that in the last decade [4, 12, 20]; however, more effort is needed to explore further what CT is, how CT can be properly operationalized in educational activities, what distinguishes CT from other kinds of thinking skills, and how CT can be incorporated within other subject domains. While the emphasis of CT education was more on the K-12 educational settings in the early days, it has gradually attracted more and more attention in higher education. In this study, CT is deemed as a skill independent of conventional programming or computing skills.

Assessment can be a means to assure the quality of education and to support students in their learning process. Examining how a competency is assessed can help us understand the operationalization and incorporation of the competency. A literature review is an intuitive method for gaining a holistic review of CT assessment in higher education. Though several existing reviews examined CT assessment from different aspects, as far as we are aware, an umbrella review that revises and summarizes knowledge accumulated on CT assessment in higher education is yet to be presented.

In this study, we synthesize existing knowledge by performing a systematic umbrella review on reviews relevant to CT assessment in higher education. Our research questions are as follows:

- RQ1 What are the features of the existing reviews? Sub-questions are listed as follows:
  - RQ1a What are the bibliographical characteristics of the reviews, such as year of publication, country of the work, and type of publication?
  - RQ1b What are the methodological characteristics of the reviews, such as type of review, principles, methodology followed for the review, and tools used for reviewing?
- RQ2 What are the explored aspects of CT assessment in the reviews? Sub-questions are listed as follows:
  - RQ2a Which constructs have been found in CT assessment?
  - RQ2b Which aspects characterize the methodology applied for CT assessment?
  - RQ2c What characterizes the contexts in which CT has been assessed?

Preliminary results of this work were presented in the work of Zhang and Specht [21]. According to the results, existing reviews hardly refer to other reviews and rarely synthesize existing knowledge which can provide the readers a comprehensive view of the field. Therefore, in this work, by investigating all existing reviews (as exhaustive as it can be), we aim to gather and synthesize knowledge to picture the field. In this study, we extend the previous work with (1) an extensive analysis of the publication features, with a discussion on the types of publication and the trends that emerge in the publications (RQ1a), (2) an extensive analysis of the methods adopted by the included reviews (RQ1b), (3) an overview of the aspects investigated on the topic of CT assessment in higher education (RQ2), and (4) a comparison and analysis of the following topics: the assessment constructs (RQ2a), the assessment methodology (RQ2b), and the assessment context (RQ2c).

2.2. Related Work

2.2.1. CT and its Assessment in Higher Education

A considerable amount of knowledge has been accumulated on the topic of CT education. Theoretical frameworks have been established over the years to facilitate CT understanding and promotion, such as Brennan and Resnick's three-dimensional framework [22], Weintrop's taxonomy of CT in mathematics and science [23], and Grover and Pea's competency framework [24], which are seemingly the most frequently adopted in the literature. Tools such as Alice, Scratch, and Bebras, have been developed for teaching, learning, and assessment of CT in different contexts [20]. Moreover, curricula have also been developed for teaching CT in different contexts [25].

It is noteworthy that CT has been mostly coupled with CS and programming, while it is regarded as a skill that can be applied to not only domains such as engineering and mathematics, but also daily life scenarios [19, 26–28]. Moreover, contributions focused more on the context of K-12 than on higher education [20]. It was only after 2014 that research on CT in higher education started to develop briskly, with a focus on operationalizing and incorporating CT in programming and CS contexts [20]. Meanwhile, the assessment of CT is highly diverse due to the different implementations and operationalizations of CT in different educational contexts [29]. Furthermore, the assessments are rarely validated and most of the work suggests a combination use of different assessment methods [4, 7, 20, 27, 30].

2.2.2. Reviews on CT Assessment in Higher Education

According to the results [21], there are 11 reviews that either fully focus on CT assessment in higher education or discuss CT assessment in higher education as one of their components. Among the reviews that mainly focus on the topic of CT assessment, De Araujo et al. investigated specifically the assessed abilities [14], Haseski et al. focused on analyzing features of paper-and-pencil data collection instruments [31], Vinu Varghese and Renumol [32] and Poulakis and Politis [33] examined the topic by studying methods and approaches appeared in the studies while Tang et al. [12] looked more holistically how CT has been assessed in studies, and Cutumisu et al. [20] and Lu et al. [7] analyzed vital features of CT assessed, with the work of Lu et al. specifically focusing on higher education [7]. In other reviews, Lyon and J. Magana [4] and De Jong and Jeuring [13] described CT assessment methods as part of the results for the investigation of key areas for future study and interventions used in higher education, respectively. Meanwhile, Taslibeyaz et al. [34] investigated both conceptual understanding and measurement approaches for CT development, and Ezeamuzie and Leung [30]] studied various operationalizations of CT in the literature, presenting CT assessment as part of their results.

2.3. METHOD

An umbrella review is a review of reviews on a topic that compiles all the evidence of existing reviews to present a high-level overview [35]. With more studies emerging on the topic of CT assessment, researchers need to have an overview of the field; to the best of our knowledge, there is not a study that provides a high-level overview. To address this gap and answer the research questions, we conduct a systematic umbrella review to investigate the topic of CT assessment in higher education.

This study adopts a systematic process for gathering data with the following steps [36]: (1) identify the scope and formulate research questions, (2) plan the review and create a documented protocol, (3) develop inclusion and exclusion criteria, (4) search and screen the studies, (5) extract and synthesize data. We conducted a narrative analysis of the literature to narrow the scope of our work and establish a documented protocol. Inclusion and exclusion criteria were developed with the expert's suggestions. The PRISMA flowchart was adopted for recording the results of the first four steps, while all other data were documented in an Excel file. Aspects of the included reviews were examined through discussions between the authors where necessary. More details about the whole process are presented in the following subsections. The principal results for the key steps are shown in Figure 2.1.

2.3.1. INCLUSION AND EXCLUSION CRITERIA

We developed four exclusion criteria and two inclusion criteria to identify the eligible reviews for further analysis. The criteria are listed in Table 2.1.

2.3.2. IDENTIFICATION AND FILTERING THE REVIEWS

DATA SOURCES AND SEARCH QUERY

Two world-leading citation databases [37] were used as main data sources to ensure the coverage of the work: Web of Science (WoS) and Elsevier journal database (SCOPUS). Google Scholar was used as an additional source to identify relevant records. We used “computational thinking” and “review” as keywords for search queries; “higher education” was left out to include all potentially relevant records. Search queries are listed as follows (first round search at 2021-08-25):

- Web of Science: TS = (“computational thinking” AND “review”)
- SCOPUS: TITLE-ABS-KEY (“computational thinking” AND “review”)

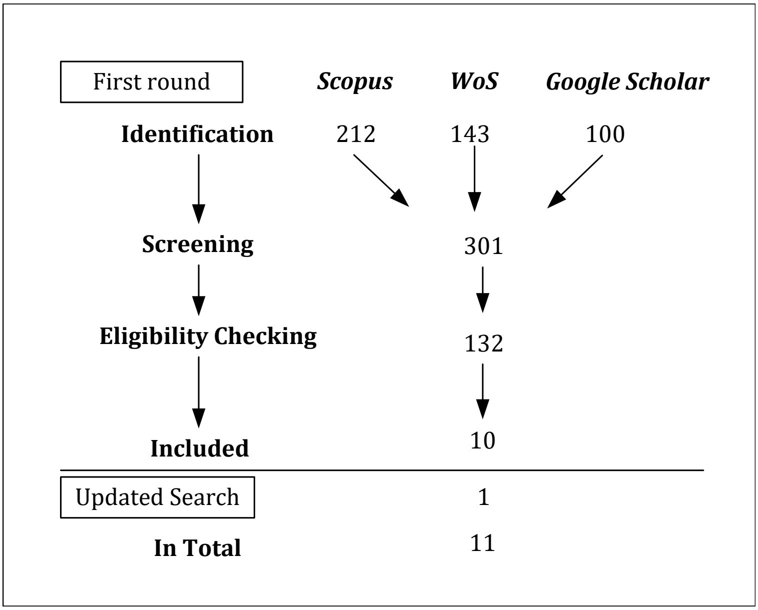


Figure 2.1.: PRISMA flowchart with results

Table 2.1.: Inclusions and Exclusions Criteria

Exclusion criteria	**Inclusion criteria**
• The study is inaccessible (The full-text article is neither available in the databases nor upon the request of the authors). • The study is not peer-reviewed. • The study is written in a non-English language. • The study is a duplicate of the existing ones.	• The type of the study must be a review. • The study contains content about the assessment of CT in higher education.

With these queries, we retrieved records in Scopus and Web of Science that contain "computational thinking" and "review" in at least one of the following fields: abstract, title, and keywords. As for Google Scholar, we examined the first 10 pages of Google Scholar results, sorted by relevancy, with the search query "computational thinking" and "review". In total, we identified 355 records from SCOPUS and WoS and the first 10 pages (100 records) of Google Scholar results for screening and eligibility checking.

Screening and Eligibility Checking

For all records identified from WoS, SCOPUS, and Google Scholar, we applied the inclusion and exclusion rules to their titles, abstracts, and keywords for preliminary filtering of the records. Before the screening stage, we removed duplicated records, resulting in 301 records. At the screening stage, by reading titles, keywords, and abstracts, we removed records that were non-English, not peer-reviewed, or inaccessible, as well as records that did not discuss CT assessment in higher education or were not reviews. This resulted in 132 records for eligibility checking. For eligibility checking, we checked the full texts to exclude the records that were not reviews or did not discuss the assessment of CT in higher education, leaving 10 reviews for data extraction and synthesis. Furthermore, an updated search on 2022-02-02 in all three databases with the same procedure resulted in one more eligible review. In total, we included a total number of 11 reviews [4, 7, 12–14, 20, 30–34]for analysis in this work.

2.3.3. Data Extraction and Synthesis

To answer our research questions, we qualitatively analyzed the included reviews with multiple features using Excel and ATLAS.ti as the data extraction and analysis tools. To answer RQ1, we examined the meta-information of the reviews and the title, abstract, and keywords of the reviews to extract bibliographical characteristics. As for the methodological characteristics of the reviews, we examined the abstract and methodological description section. Then, to answer RQ2, we analyzed the reviews with an emphasis on the abstract, introduction, method, and result sections. The analysis of the information extracted for research questions RQ1a and RQ2a to RQ2c followed a bottom-up approach; we first identified and marked all relevant information and then aggregated and synthesized that information according to the research questions.

2.4. Results and Analysis

2.4.1. RQ1 Bibliographical and Methodological Characteristics of the Reviews

RQ1a – Bibliographical Characteristics of the Reviews

We extracted the following features of the reviews from the meta-information as bibliographic characteristics: year of publication, region of the work, and publication outlet. The results are shown in Table 2.2.

The results show that most reviews were published within the last five years and no reviews were found before 2016. This increase in the number of reviews indicates a growing interest in the topic. As we look at the region of the reviews, we can assert that the contribution to this research topic is international, with the United States, Turkey, and Canada being slightly more active. Meanwhile, the contributions 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.

Table 2.2.: Bibliographical features of the reviews on CT assessment in higher education

Characteristics	**Results and Frequency**
Publication year	2016 (1), 2019 (2), 2020 (4), 2021 (3), 2022 (1)
Region of the first author	Brazil (1), Canada (2), Greece (1), Hongkong (1), India (1), The Netherlands (1), the United States (2), Turkey (2)
Publication outlet	Journals (7), Conferences (3), Book chapter (1)

RQ1b – Methodological Characteristics of the Reviews

For all reviews, we identified the following methodological characteristics: type of review, principles and methodology adopted, searched databases, inclusion and exclusion criteria (e.g. education level), quality control, and focus of review. The results are shown in Table 2.3.

To start with, for the type of reviews, we found five systematic reviews while the others were scoping reviews, narrative reviews, and a systematic mapping study. Regarding the principles and methodologies applied in the reviews, even though 10 out of 11 reviews followed or referred to existing methods or guidelines, hardly any review thoroughly explains the choice of the adopted methods. In terms of the searched databases, every review used two to six databases as data sources, and there were in total 15 databases identified as data sources, being either domain-specific databases or general indexing databases. Among the domain-specific databases, ERIC was the most used database, followed by the general indexing database Scopus. Moreover, the eligibility of the retrieved items was checked against selection criteria in all reviews, with several of the reviews specifying both the inclusion and exclusion criteria. Regarding the review process, most of the reviews described how discrepancies in the selection of reviewed studies and the collection and analysis of information were resolved, while the rest did not indicate that topic.

Last but not least, while sharing some similarities, the scope of the reviews differ from each other, meaning that each review focuses on different aspects. Among all reviews, only Lu et al. [7] exactly scoped their review on the topic of CT assessment in higher education by examining empirical evidence. For the reviews that limited their scope to higher education, one provided a general picture of CT in higher education [4] and another [13]] studied CT interventions in higher education from empirical experiences, with assessment of CT being only a part of the investigation. For the rest of the reviews with which the scope was not confined to higher education context, they investigated the assessment of CT across all educational levels including assessment of CT [14, 33], data collection instruments for CT assessment [31], empirical experiences of CT assessments [12, 20], empirical experiences on the development of CT [34], empirical experiences of CT practices [30], and assessment methods and interventions for developing CT [32]. Additionally, seven out of 11 reviews investigated empirical evidence which collected information through observation and documentation of certain behaviors and patterns or an experiment.

Table 2.3.: Methodological characteristics of the examined studies on CT assessment in higher education

Characteristics	Results and Frequency
Type of review	Systematic review (5) Scoping review (3) Narrative review (2) Systematic mapping study (1)
Principles and methodologies	Adopted existing methods or guidelines (10) No specification on methods used (1)
Searched databases	ACM/ACM Digital Library (6) EBSCO (1) ERIC (9) Engineering Village: Inspec (1) Engineering Village: Compendex (1) Google Scholar (2) IEEE Xplore (5) PsycINFO (1) ProQuest (1) ScienceDirect (5) Scopus (6) Springer/SpringerLink (5) Web of Science (3)
Selection criteria	Included structured description of inclusion criteria and exclusion criteria (9) Described some criteria for the selection of the studies (2)
Quality of the review process	Described how discrepancies were addressed (8) Not specified (3)
Scope of the review	CT assessment in higher education as: - Dominant topic (1) - Non-dominant topic (10)
	Type of Evidence Included - Empirical Studies (7) - Not-specified (4)

2.4.2. RQ2 – OVERVIEW OF ASPECTS OF CT ASSESSMENT EXAMINED IN THE REVIEWS

Table 2.4 presents an overview of aspects investigated on CT assessment in higher education concerning the RQ2 and its sub-questions.

Regarding assessment constructs, a total of eight reviews examined assessed constructs and their characteristics regarding the skills, and competencies. Of the other three reviews, two examined the definition of CT with which the assessed constructs can be deduced according to the constructive alignment theory [38]. As for assessment methodologies, except for the work from De Jong and Jeuring

2

Table 2.4.: Aspects investigated on CT assessment in higher education

Themes	References
Assessment Object/Construct (8)	[7, 12–14, 20, 31, 33, 34]
Assessment Methodology (10)	[4, 7, 12, 14, 20, 30–34]
Assessment Context (11)	[4, 7, 12–14, 20, 30–34]

[13], all reviews examined perspectives relevant to the delivery of the assessment, namely instruments and tools developed for assessment and their characteristics, categorization of assessment methods, and quality indicators for the assessment methods. In terms of assessment contexts, all reviews investigated this topic with different focus on the following perspectives: academic domain, education level, educational setting, and intervention. Details are presented in the following subsections.

RQ2a – Characteristics of the Assessed Constructs

Table 2.4 shows that eight out of 11 reviews examined the assessed constructs. Regarding the method applied for studying assessed constructs, as presented in Table 2.5, by applying (a) Sullivan and Heffernan (2016)'s categories of first-order and second-order learning outcomes regarding the relationship between CT and other subject domains [39] and (b) McMillan (2013)'s categories of cognitive constructs and non-cognitive dispositions and skills [40], Tang et al. [12] examined the relationship between the assessed constructs and the subject domain. For the remaining set of reviews, Cutumisu et al. [20] and Lu et al. [7] utilized an existing CT framework as a reference to identify the assessed constructs, while the others clustered and synthesized similar constructs that emerged in their investigation [13, 14, 31, 33, 34].

Table 2.5.: An overview of the methods and analysis applied for the investigation of the assessed constructs

Method	References
Apply an Existing Classification Scheme (1)	[12]
Utilize a Existing Framework (2)	[7, 20]
Natural Clustering of Constructs Emerged from the Study (5)	[13, 14, 31, 33, 34]

As for assessed constructs and their categorization, Cutumisu et al. [20] and Lu et al. [41] mapped the assessed constructs to Brennan and Resnick's three-dimensional framework [42] and a hybrid framework inferred from three frameworks [22–24], respectively. Though both frameworks sketched CT competencies as a three-dimensional framework, including CT concepts, practices, and perspectives, the hybrid one was claimed to be more generic and independent from subjects allowing a broader coverage of CT skills. The other five reviews presented: (1) only the overarching categories of assessed constructs that provide a high-level categorization of assessed constructs [13, 34]; (2) only the assessed constructs [14, 33]; (3) both the constructs and the overarching categories [31]. Taslibeyaz et al. [34] and De Jong and Jeuring [13] identified a total number of six distinct categories, including attitude towards CT, attitude-motivation, CT knowledge, CT skills, problem-solving skills, and programming skills without specification on the differences of categories. Meanwhile, De Araujo et al. [14], Haseski et al. [31], and Poulakis and Politis [33] identified five categories, namely affective achievements towards CT, cognitive achievements towards CT, CT skills/abilities, CT concepts, CT patterns, and enumerated the underlying constructs.

It should be noted that some constructs were classified into distinct categories in different reviews as schemes used for classification varied from one review to another. Moreover, some categories nearly shared the same name, such as CT skills versus CT skills/ability, and attitudes towards CT versus attitude-motivation. However, in this work, we did not merge them since information on their meaning is insufficient at the current level of investigation. To sum up, our approach involved merging constructs or categories deemed identical, either because they align with the same definition or are presented identically.

With the constructs considered the same being merged, we identified a total of 120 unique assessed constructs. As there are too many to enumerate, by adopting the hybrid framework used in Lu et al. [7]'s work, we present the assessed constructs that appeared in at least 3 reviews in 2.6. Example definitions of these constructs are provided accordingly.

Table 2.6.: CT assessment constructs appearing at least three times in the included 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 [24].
	Data/data analysis/data collection, data analysis/data representation (f=5)	Including storing, retrieving, updating values as well as analyzing, and visualizing data [22, 23].
	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 [24].

	Logic/logic and logical thinking (f=4)	Logical thinking involves analyzing situations to make a decision or reach a conclusion about a situation [24].
	Critical thinking (f=3)	Not found.
	Evaluation (f=3)	Solutions to problems are evaluated for accuracy and correctness concerning the desired result or goal [24].
	Pattern/pattern recognition (f=3)	Pattern recognition in CT could result in a definition of a generalizable solution that can utilize automation in computing for dealing with a generic situation [24].
	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 [23, 24].
	Problem-solving (f=4)	Not found.
	Modularity/modularizing modeling (f=3)	Building something large by putting together collections of smaller parts, is an important practice for all design and problem solving [22].
	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 [23, 24].
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 [24].

	Collaboration and cooperation (f=3)	Perspectives about working with CT skills in a collaborative or cooperative format [24].

RQ2A – CHARACTERISTICS OF THE ASSESSED METHODOLOGY

As Table 2.4 shows, 10 out of the 11 reviews studied different characteristics of the assessment methodology, mainly assessment methods, assessment tools, assessment instruments, and assessment quality indicators.

As presented in Table 2.7, four reviews presented classifications of assessment methods. Lu et al. [7] categorization resembles the one used by Cutumisu et al. [20] since it further investigated topics from Cutumisu et al. [20]'s review; however, it focused more on higher education. Both of these two reviews identified the following types of assessment:

- block-based assessments – solving programming problems regardless of syntax by using programming blocks in block-based programming environments such as Scratch;
- CT skill written tests – using generic forms such as constructed response questions or multiple-choice questions to assess CT skills, e.g. Computational Thinking Knowledge test (CT Knowledge test);
- self-reported scales/surveys – mostly concerned with assessment of CT perspectives which includes inter- and intra-personal skills such as communication, collaboration, or questioning;
- using multiple forms of assessment – combining different types of assessment to gain a more holistic understanding of student understanding or mastery of skills, for example, the combined use of the CT Knowledge test and the block-based assessment in Scratch.

Additionally, Cutumisu et al. [20] identified robotics/game-based assessments as a unique category to refer to assessments based on robotic tangible tasks or artifacts produced in game-based assessments such as AgentSheets. Meanwhile, Lu et al. [7] identified three more categories compared to Cutumisu et al. [20]'s work:

- 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, exams, projects, and assignments.

Besides that, Lyon and J. Magana [4] categorization resembled Lu et al. [7] and Cutumisu et al. [20]' reviews, while being more generic. We noticed that, while these three reviews categorized assessment methods based on the type of assessment

2

Table 2.7.: Type of assessment methods identified from the reviews

Assessment methods	**References**
Block-based assessments	[7, 20]
Knowledge/CT skill written tests/skill tests	[4, 7, 20]
Self-reported scales/survey	[4, 7, 20]
Robotics/game-based assessments (tangible tasks)	[20]
Combinations/using multiple forms of assessment	[4, 7, 20, 33]
Text-based programming assessment	[7]
Course academic achievements of CS courses	[7]
Interviews or interviews and observations	[4, 7]
Assignment/course grades	[4]
Artefacts (classroom/students)	[4]
Problems external to class	[4]
Using specific programming environments	[33]
Using CT assessment criteria/psychometric tools	[33]

instrument and the medium used for assessment, Poulakis and Politis [33] classified the assessment methods based on the characteristics of the medium. However, none of these classification schemes was constructed on a theory basis but was rather established by summarizing the patterns that emerged in their studied sets of papers.

Additionally, some reviews presented additional aspects of assessment methodologies of CT, including assessment tools and assessment instruments used for the operationalization and delivery of assessment, and assessment quality indicators for controlling the quality of the assessment method. Table 2.8 shows the reviews that investigated those topics, following the original categorization naming presented in the reviews. It is worth mentioning that no reviews explicitly describe what differentiates instruments from tools.

Regarding assessment tools, Lu et al. [7] and Cutumisu et al. [20] identified and reported them in a table, while the other reviews either examine the characteristics of the tools or categorize the tools according to certain characteristics and exemplify tools of those categories in text. Characteristics of the tools being studied include the programming language necessary for the tools [20], the type of the tool [12, 31, 34], and the functionality of the tool being either summative or formative [34].

As for assessment instruments, characteristics identified include the delivery format being either automatic or non-automatic [7, 20], computer-based or paper-based [7, 20], the type of skills included being cognitive or non-cognitive [7, 20], the number of items and constructs included in the instrument [31], the type of assessment tasks in the instrument such as coding projects [20].

Moreover, two reviews systematically grouped the assessment quality indicators for evaluating the quality of the assessment methods regarding the validity, the reliability, or both the validity and the reliability [20, 31].

Table 2.8.: Additional characteristics of the assessment methodologies identified from the reviews

Aspects	Description	Reference
Assessment Tools (6)	Aspects including the name of the tool and the characteristics of the tools, the programming language, type of assessment tool, the functionality of the tool	[7, 12, 20, 31, 32, 34]
Assessment Instruments (5)	Aspects Including the type of instruments and the characteristics of instruments, e.g. delivery format, type of skills included, type of instruments, number of factors and items included in the instrument	[7, 14, 20, 30, 31]
Assessment Quality Indicators (2)	Validity and reliability indicators for the assessment methods.	[20, 31]

RQ2c – Characteristics of the Assessment Context

As shown in Table 2.4, all reviews discussed assessment contexts. However, only a few examined every aspect of the assessment context. Table 2.9 presents the major aspects, being academic domain, education level, educational setting, and intervention that can affect the quality and the design of the assessment.

Table 2.9.: Characteristics of assessment context

Aspects	Description	Reference
Academic Domain (7)	Concerned with an investigation into academic disciplines, programs of studies, or subject matter for the assessed group of users.	[4, 7, 12, 13, 20, 30, 34]
Education Level (11)	Concerned with the level of education for the assessed group of users.	[4, 7, 12–14, 20, 30–34]
Educational Setting (5)	Concerned with the type of educational activities that the assessed group of users were involved in.	[7, 12, 14, 20, 30]
Intervention (6)	Concerned with the actions taken for the development of skills and or their corresponding characteristics.	[4, 13, 20, 30, 32, 34]

In total, seven reviews reported the academic domain of the assessment receiver, with different levels of detail; meanwhile, the others mentioned it as supplementary information or presented it in an unstructured and uncomprehensive manner. De Jong and Jeuring [13] presented a list of academic disciplines, while the others presented either information about the coursework within a study program or a categorization of the coursework/the study program. Though there are various schemes for categorizing the academic domain or the coursework, the reviews mainly distinguished the academic background according to the relevance of the study program to CS, Science, Technology, Engineering, and Technology [12, 13, 30, 34]. Furthermore, all reviews revealed that most of the contributions on CT were within the scope of STEM disciplines, CS and Programming Education, with an increase in the number of studies regarding Non-STEM and non-CS majors and subjects in recent years. This finding complies with the conclusions in several of the included reviews.

Regarding the education level, all reviews included it, with some exhaustively listing all grade levels that indicate the year of study, whereas the others presented it at a higher level without detailed information on the year of study. For reviews that focused on higher education [4, 7, 13], Lu et al. [7] grouped and tabulated the grade level according to the year of study, while the other two reviews regarded undergraduate as a category that covers all undergraduate years. The rest of the reviews did not limit the scope to higher education, most of them made the distinction between the K-12 and undergraduate context, with some providing finer categorizations. Some reviews analyzed the distribution of education levels of the included studies to gain insights into research trends [14, 31], while some others such as Taslibeyaz et al. [34] examined the tools or assessment methods used in certain levels of education.

Some reviews illustrated the assessment context with more details, including the educational settings and the interventions. Educational settings provide general information about the characteristics and the form of educational activities formal or informal, representing in-school activities such as lectures or workshops, and after-school activities such as coding clubs. As shown in Table 2.9, five reviews examined the educational settings. Meanwhile, for those who presented no explicit indication, the educational setting of the study can be deduced from the intervention type, such as the reviews from Lu et al. [7] and Cutumisu et al. [20]. In total, six reviews illustrated the intervention categories in different manners. However, information on interventions covers some additional aspects, such as the intervention duration, the pedagogical approach, and the tools used in the intervention [13, 31].

2.5. Discussion

2.5.1. RQ1 Towards a More Rigorous Review Method and More Exploration of Different Aspects CT Assessment in Higher Education

The findings on the bibliographical and methodological characteristics of the reviews show an increased focus on exploring CT assessment in higher education. Most reviews are systematic reviews that adopted existing review methodologies or guidelines and were published in journals or, less commonly, in conferences. However, as far as we are concerned, none of the reviews investigated comprehensively existing reviews or reasoned on the choice of the applied methodology. Though it can be argued that CT education, specifically in the context of higher education, is a relatively young field, due to the number of studies and reviews that have been conducted, we contend that an umbrella review is necessary before conducting a systematic review to properly locate potential gaps in the field.

Since studies of CT fall under a multidisciplinary scope, the searched databases varied from generic indexing databases such as Scopus to domain-specific databases such as ERIC. With these results, for reviews that aim to examine studies conducted on the topic of CT, it is advisable to include generic databases, as well as domain-specific databases to ensure the maximum exhaustiveness of the search. The retrieved articles, in most of the reviews, were filtered with explicit inclusion and exclusion criteria; however, the quality of the review process was mostly justified by explaining the method used for resolving discrepancies. To have a more rigorous method for reviews, we contend that it is worth discussing the methods to assure the coverage of retrieved data as exhaustively as possible, including the selection of databases, the establishment of inclusion and exclusion criteria, and the resolution of discrepancies.

Topic-wise, only one review investigated CT assessment in higher education, while the rest either studied the topic as complementary aspects of the main focus or with an emphasis on other aspects of higher education. Since most of the reviews aim at discussing CT assessment in higher education as a complementary topic, it should be noted that the insights in this umbrella review may also concern mixed findings, including existing knowledge in K-12 settings as well as in higher education, and other aspects of CT education. Specifically, only some reviews discussed results on assessment constructs, assessment methodologies, and assessment contexts; more effort is necessary to enrich the field from all aspects. However, we consider our findings as a reference for the design, scoping, and discussion of future studies.

2.5.2. RQ2 Assessment Design - Assessment Constructs, Methodologies, and Contexts

For RQ2, we investigated the assessment constructs, assessment methodologies, and assessment contexts of CT assessments explored in the reviews. At a high level, those three aspects were studied in more than two thirds of the reviews, suggesting that they can be crucial components to consider when designing assessments. For those three aspects, a finer level of investigation provides the reader with more details on the potential factors to consider for assessment design.

First of all, we identified 120 unique constructs, with some being clustered from included studies and the others [7, 20] drawing on Brennan and Resnick's CT framework [22] and the hybrid framework [22–24]. It appears to be that the hybrid framework covers most of the frequent constructs, which indicates a certain level of consensus on the assessed constructs. However, it should be stressed that these results were based on reviews that also included studies in the K-12 context. Moreover, none of the studies examined whether certain constructs appeared more often or were considered more appropriate to be assessed at a certain educational level. Therefore, we argue that there is still room for exploring which constructs should be included in various kinds of assessment within specific contexts.

Secondly, in terms of assessment methodology, we found various assessment methods, assessment tools, assessment instruments, and assessment quality indicators. Methods are grouped and presented differently in four reviews and the reviews promoted a combined use of different assessment methods. Besides that, some reviews investigated extensively the tools and instruments and their characteristics, depicting the delivery of assessments and the medium for assessment less abstractly and more intuitively. Most instruments were computer-based with non-automatic scoring, indicating a preference for computer-based assessment over paper-based assessment. Moreover, most tools assess both cognitive and non-cognitive skills, which complies with the idea of a more comprehensive understanding of one's competency level. Nonetheless, none of the reviews systematically examined the tools or instruments regarding the level of skills assessed. Further mapping and categorization of tools and instruments and their intended assessment level of skills is necessary for proper progressive assessment design. Last but not least, most reviews did not examine the validity and reliability of the assessment, appealing for more attention on this topic.

Lastly, regarding the assessment context, we examined four aspects: academic domain, education level, educational setting, and intervention. We found an increased number of studies integrating CT into various disciplines, with a growing attention to non-CS majors. Studies were mostly conducted in a formal educational setting and assessments were mostly conducted with entry-level or lower-level students within a course or curriculum. However, there is no consensus on which assessment is more proper for different education levels and students from different backgrounds within the context of higher education. Furthermore, though reviews examined characteristics of the interventions applied for the development of CT skills, there was hardly any focus on the relationship between intervention and assessment design. Future studies on assessment could pay attention to constructive alignment, meaning aligning the assessment with the learning objectives and the interventions used in the education scenarios.

2.6. Conclusion and Future Work

To gain insights on the topic of CT assessment in higher education, this study systematically reviewed 11 reviews from the following perspectives: bibliographical features of the reviews; methodological features of the reviews; constructs/objects being assessed; applied assessment methodologies; assessment contexts which provides information about the assessed participants and the learning activities. The major findings of this study are that (1) there is an increase in the number of studies on the topic of CT assessment in higher education, while more efforts are still needed for exploring different aspects of the topic to further develop the field; (2) there is little discussion on the selection of the review methodology and the selection of databases; (3) there is a plethora of constructs available to be included in assessments; (4) there exists no systematic investigation of the tools or instruments regarding the level of skills being assessed; (5) there are no guidelines for assessing certain constructs with certain methods in different contexts; (6) constructive alignment theory is hardly applied for aligning the interventions with the learning objectives and assessments.

Regardless of the findings presented above, there are still questions to be explored such that we gain more understanding and knowledge for assessing CT in the context of higher education. In terms of assessment constructs, it can be further examined which are necessary to include in assessments in higher education and different contexts; towards establishing principles for assessment design, the variation of applied assessment constructs can be studied for diverse assessment contexts; finally, in terms of assessment context, a remaining open question is whether assessment conducted in different contexts should vary and how this variation would affect assessment quality.

3

A Systematic Review of Empirical Studies on Computational Thinking Assessment in Higher Education

We need guidelines for different skill levels of computational thinking to support competency tests.

Tedre and Denning., 2016

Efforts have been devoted to several aspects of Computational Thinking (CT) education; however, CT assessment in higher education remains one of the challenges in the field because of its complexity and ambiguity. Though existing reviews strive to provide insights into the topic of CT assessments, we recognized that a review to investigate potential aspects to be considered for assessment design is yet a gap in the realm of CT education. Therefore, this review aims to outline the developments of CT assessment in higher education and the potential aspects of assessment design by examining the purpose, the constructs, the approaches, and the contexts of the assessment. Following a systematic review method, we reviewed 47 empirical studies published between 1986 and 2021. Findings reveal that most assessments were conducted with a summative purpose in STEM fields at the level of undergraduates, assessing CT constructs in various naming and categorization schemes, with mainly existing assessment tools or new ones designed by the authors. We suggest future work to study further (1) the exploration of the use of formative assessment to understand the CT problem-solving process and (2) the exploration of the design of assessment with an interdisciplinary approach to link different perspectives in a holistic CT assessment approach.

3.1. INTRODUCTION

3.1.1. DESIGN OF COMPUTATIONAL THINKING IN HIGHER EDUCATION

Computational Thinking (CT) education has developed rapidly since Wing promoted the idea of CT for all in 2006 [1]. Until now, there has been no consensus on the definition of CT [12] and the need for shared definitions has been especially highlighted by educators [8]. In practice, educators and educational researchers operationalize and assess CT differently.

Though researchers worldwide have been striving to develop and validate assessment tools, gaps exist in the field. First of all, most of the tools focus on K-12 education, whereas higher education has gained only more attention in recent years [8, 12, 20, 41]. Second, only a few standardized tools exist in the field due to the complexity and ambiguity of CT itself. Moreover, one can hardly find any guidelines on designing CT assessments. As such, it remains a challenge to design a CT assessment, especially at the level of higher education and different disciplines.

3

Following the idea that the design of the assessment is essential to successfully assess skills, in this study, we attempt to establish common ground for constructing a thoroughly planned assessment by aligning assessment features such as assessment purpose, the assessed items and constructs, its context, the delivery method, and its validity and reliability [43]. The focus of this work is a) the discussion of researchers and educators on shared approaches for CT assessment and b) common ground for developing assessment methods for CT.

3.1.2. RELATED WORK

To set the scope of this review, we conducted an umbrella review [21] and identified 11 reviews that investigated the characteristics of CT assessment in higher education. Findings suggest that CT assessment in higher education received less attention compared to CT assessment in K-12 education [8, 12–14, 20, 30–34, 41], and only about half of the reviews focus on empirical studies [8, 12, 30, 34, 41].

All reviews focused on various aspects of the assessment of CT. For reviews that focus on more than higher education, they studied the critical features of existing CT assessments, with emphasis on different aspects such as the educational level, the assessment tools, and the features of the participants [12, 14, 20, 30–34]. Only three reviews focus merely on higher education; De Jong and Jeuring conducted a scoping review to investigate CT interventions used in higher education and reported assessment methods [13], Lyon and Magana identified CT assessment methods while exploring the development of CT in higher education [8], and Lu et al. conducted a scoping review to identify trends of empirical research and key features of CT assessment instruments [41].

In a nutshell, existing reviews have examined CT assessment in higher education with different emphasis on the following perspectives: the assessment context concerning aspects like academic discipline, the educational setting, and the educational level; the assessment method that features aspects of assessment delivery; the assessment constructs that relate to what to be assessed and their characteristics; the reliability and validity of the studies as well as the assessment tools. The review that is most relevant to this work is the one from Lu et al. [41], a scoping review that studied 33 journal papers to identify trends of empirical research and key characteristics of CT assessment instruments in higher education. In addition to existing reviews, in this work, we attempt to systematically analyze CT assessment, with emphasis on potential features for assessment design, to answer specifically the following research questions for a higher education context:

RQ1 For what purpose is the CT assessment conducted?

RQ2 Which underlying constructs are assessed?

RQ3 Which methods have been used for the assessment?

RQ4 In which context and with what kind of study design has CT been assessed?

3.2. Method

This review is setup based on the framework from Arksey and O'Malley (2005) and follows the PRISMA extension for scoping literature reviews (Tricco et al., 2018). We followed the following steps: 1) formulating the research question, 2) identifying relevant studies, 3) selecting the eligible studies, 4) charting the data, and 5) collating, summarizing, and reporting the results.

3.2.1. Information Sources, Identification and Selection of the Studies

To ensure the coverage of the literature, we used the following two citation databases [37]: Web of Science (WoS) and Elsevier journal database (SCOPUS). We select the final set of papers in three steps: identification, initial screening, and eligibility checking. First, based on the scope of the review, the search keywords for identification of the papers are:

- "Computational Thinking" OR "Programming Education", AND
- "Assessment" OR "Indicator" OR "Intelligent Tutoring" OR "Feedback"

Then, the search keywords were used for search queries in the targeted databases (search date: 2021-01-22). The results are shown in Table 3.1.

The results were exported to a spreadsheet with the title, abstract, author's keywords, and other meta-data from each database. Those spreadsheets were then collated in the same format in a spreadsheet for screening. During the screening process, exclusion criteria were applied to exclude papers based on the information in the spreadsheet. The exclusion criteria were:

- The paper that is a duplicate or an older version/edition/release of another document.

Table 3.1.: Search queries and number of results in WoS and Scopus databases

Database	Results	Search Query
WoS	351	**#1:** TI=("Computational Thinking" OR "Programming Education") OR AB=("Computational Thinking" OR "Programming Education") OR AK=("Computational Thinking" OR "Programming Education") **#2:** TI=("Assessment Indicator" OR "Assessment" OR "Indicator" OR "Intelligent Tutoring" OR "Feedback") OR AB=("Assessment Indicator" OR "Assessment" OR "Indicator" OR "Intelligent Tutoring" OR "Feedback") OR AK=("Assessment Indicator" OR "Assessment" OR "Indicator" OR "Intelligent Tutoring" OR "Feedback") **#3:** #1 AND #2
Scopus	693	TITLE-ABS-KEY("Computational Thinking" OR "Programming Education") AND TITLE-ABS-KEY("Assessment Indicator" OR "Assessment" OR "Indicator" OR "Intelligent Tutoring" OR "Feedback")

- The paper that is inaccessible. Some papers could not be obtained even after contacting the author(s).
- The paper is not written in English.
- The paper is not published in a peer-reviewed journal or peer-reviewed proceedings of a conference.
- The paper is not a full paper, e.g., a panel announcement, work-in-progress paper, workshop overview, etc.

Lastly, by reading the body of the articles, the remaining papers were checked according to the following inclusion criteria:

- The study must be an empirical study. CT must be empirically assessed, and papers discussing theoretical frameworks proposed for CT assessment were thus excluded.
- The educational context must be higher education. The goal of this study is to identify the assessment of CT in higher education.
- The content must contain the assessment of CT. The study must discuss at least one characteristic or property of assessment, such as assessment purposes, methods, and constructs.

3.2.2. Coding and Information Extraction

Eligible studies were bundled and imported to ATLAS.ti[1]. We followed the adapted three-stage content analysis (Fraenkel et al., 2011) for the coding and extraction of information. First, a coding scheme, initially with categories of information to be coded, was determined through discussions between the authors to extract information systematically. Then, in two iterations, the coding scheme was refined between all authors and used by the first author to code the eligible papers.

3.3. Results

Figure 3.1 presents an overview of the identification and selection process discussed in Section 3.2. Database search resulted in 1044 studies, including 286 duplicates. From the remaining 758 studies, a significant number of studies were excluded following the exclusion criteria, resulting in 260 items for detailed eligibility examination. Through eligibility checking, 47 studies were included for analysis.

3.3.1. RQ1: Assessment Purposes

According to Krathwohl [44], there are three types of assessment purposes: formative, summative, and a combination of both summative and formative. As shown in Table 3.2, most of the studies used assessment for a summative purpose. In total, 40 studies implemented assessment for a summative purpose, aiming to provide specific learning performance results. For example, Altanis and Retalis [45] used Dr.Scratch and the CT journal to analyze the students' average CT scores to evaluate the effectiveness of an assessment framework. As for the rest, four studies used assessment for a formative purpose, and three studies adopted a combination of both. Studies that use formative assessment usually come with immediate feedback to support students in their learning process. For example, Krugel and Hubwieser [46] used quizzes after video teaching materials on a MOOC platform to provide learners with feedback, and Kafura et al. [47] provided students with contextualized assistance in Blockly.

Table 3.2.: Categorization of results over assessment purpose (RQ1)

Category	Sub-codes	Reference
Assessment Purpose	Formative (4)	[48] [47] [46] [49]
	Summative (40)	[50] [45] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88]
	Both (3)	[89] [90] [91]

[1] ATLAS.ti: The Qualitative Data Analysis & Research Software

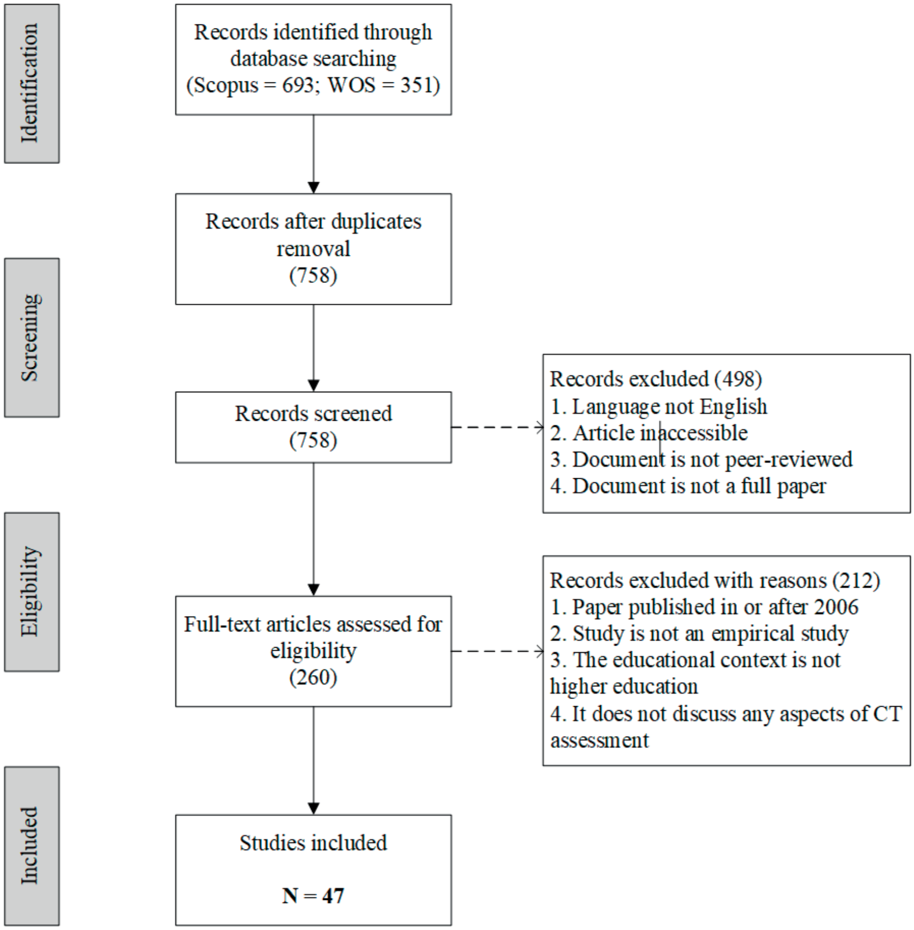


Figure 3.1.: Results for the identification and selection of the studies (PRISMA chart)

3.3.2. RQ2: ASSESSMENT CONSTRUCTS

Three aspects of assessment constructs were examined: the type of constructs, the use of scales for measuring mastery levels, and the adoption of specific frameworks.

As shown in Table 3.3, we identified eight constructs. For studies that explicitly mentioned the type of constructs they assessed, we followed their naming convention: CT affective outcomes (perceptions, attitudes, and perspectives on CT), CT competency, CT concepts, CT knowledge, and CT practices. For the rest of the

Table 3.3.: Categorization of results on assessment construct (RQ2)

Category	Sub-codes	Reference
The Type of Constructs	Computing and Programming (2)	[54] [87]
	CT affective outcomes (10)	[89] [56] [47] [62] [68] [76] [77] [78] [84] [85]
	CT competency (1)	[81]
	CT concepts (4)	[45] [89] [72] [81]
	CT knowledge (2)	[58] [64]
	CT practices (3)	[45] [89] [59]
	CT skills and others (25)	[50] [51] [48] [58] [59] [60] [47] [61] [62] [63] [65] [90] [68] [69] [70] [71] [74] [75] [79] [80] [83] [91] [84] [85] [86]
	N.A.(None-applicable) (11)	[52] [53] [55] [57] [46] [66] [67] [73] [49] [82] [88]
Other Aspects	The Use of Scales (6)	[45] [89] [59] [65] [71] [81]
	The Adoption of Specific Frameworks (13)	[45] [89] [57] [59] [65] [69] [70] [71] [72] [77] [81] [84] [85]

studies, except for those that did not describe the assessed constructs (belong to the N.A. category), constructs are categorized as CT skills and others.

For the first two categories on the type of constructs, only two studies fall into the first category; Cabo and Lansiquot [54] assessed students' understanding of procedural and object-oriented programming concepts, and Thangavelu et al. [87] assessed students' programming tasks either from Scratch or Flowgorithm. As for the second category (CT affective outcomes), 10 out of 47 studies assessed the attitude, perceptions, and perspectives of CT. For example, Pulimood et al. [77] investigated students' affective outcomes through a four-Likert scale questionnaire over questions derived from Wing's definition of CT [92] and ABET's General and ProgramSpecific (CS) Students' Outcomes Criteria for Accrediting Computing Programs [93]; Kafura et al. [47] conducted group observations and interviews to examine students' views on CT; Choi [56] implemented questionnaires to measure learners' perceptions and effectiveness of CT, programming, and problem-solving.

For the rest of the categories, over half of the studies assessed constructs that fall under one of them. For example, Romero et al. [81] measured CT competency with the #5c21 model, which consists of six components, including CT concepts. Since Brennan and Resnick's [22] framework was commonly adopted, studies mainly examined CT practices and concepts together [45, 89]. For the two studies that assessed CT knowledge, Lee and Cho [64] implemented a questionnaire to measure students' knowledge of pattern analysis, condition comparison, abstraction, automation, and algorithm design, while Francisco and López [58] chose items from Computer Olympiad Talent Search and the contest "about informatics and computational fluency school age" UK Bebras. A total of 25 studies fall into the category of CT skills and others, with some explicitly mentioning the assessed constructs as CT skills and others mentioning them as skills assessed in a CT educational activity. For example, Park et al. [75] measured students' ability to read code and reason about the hierarchical relationships between different blocks, Takemata et al.[86] measured one's ability to abstract problems and develop one's thoughts and Sondakh et al. [84] measured technical knowledge as the non-affective CT skills.

Regarding the use of scales and the adoption of frameworks, six out of 47 studies used scales, and 13 out of 47 studies adopted a specific framework. For studies that used a scale, while Leela et al. [65] reported a table with a list of assessed constructs, the scoring criteria and scores for different mastery levels, the rest either provide a scoring range or the required mastery level for each assessment item. As for studies that adopted frameworks, most either adopted an existing framework, such as Brennan and Resnick's framework [22] or designed one based on the specific educational setting, such as the #5c21 model in Romero et al.'s [81] study.

3.3.3. RQ3: Assessment Methods

Table 3.4 presents features of assessment methods from four aspects: the assessment perspective, the assessment format, the assessment method, and the assessment tool.

Regarding the assessment perspective, the teacher assessment was dominantly used. The teacher assessment usually comes with rubric grading with given criteria [56, 71, 83, 90]. Only one paper implemented peer assessment with which students were asked to star rate other students' projects [48]. As for the self-assessment, around one-fifth of the studies adopted it, primarily with Likert-scale surveys [62, 76, 78, 84, 85]. A limited number of studies used multiple types of assessment; for example, an objective CT score (OCT) scale was scored by others, while a subjective CT score (SCT) scale was scored by oneself [55].

As for the assessment format, most of the studies adopted a manual assessment, while a few used an automatic assessment or a combination of both. Manual assessment can be found in different scenarios, such as evaluating a report of a student's project [88] or evaluating students' program artifacts and projects [48, 56, 71]. In contrast, the automatic assessment focuses on the automatic assessment of programming and modelling assignments or selected-response tasks [46, 49, 51, 53]. For example, Mishra and Iyer [72] adopted rubrics to manually assess students' performance over problem generation tasks using CT concepts, Krugel and Hubwieser [46] automatically graded students' programming assignments to provide feedback to students, and Altanis and Retalis [45] adopted both formats by using Dr.Scratch to automatically assess student's code and CT journal to assess students' CT practices manually. In terms of assessment methods, traditional tests and exams were mostly implemented, followed by questionnaires and surveys, journals, and interviews. One significant category is labeled as miscellaneous, which includes using tasks, assignments, projects, and design and implementation as assessment methods. These methods are not further categorized as they normally have more varieties and more dynamic features regarding the contexts.

For the assessment tool, around two-fifths of the studies directly adopted existing tools, around two-fifths of the studies developed their tools for assessment, nearly one-fifth of the studies do not mention or describe in detail the origin of the tools, and very few studies adapted from existing tools for assessment. First, Pulimood et al. [77] adapted the existing ABET's General and Program-Specific (CS) Student Outcomes Criteria for Accrediting Computing Programs ABET Computing Accreditation Commission [93] and Wing's definition of CT [92], and Choi [56] modified an evaluation rubric presented in study to achieve the goal. Examples of existing tools directly used in the studies are automatic testing software [90], Dr.Scratch [45], CTPA [63] and Computational Thinking Test (CTT) [55]. Then, tools that were originally designed, for example, are the #5c21 competency model [81], which consists of six components designed by the authors, Digital ink [51] that captures learners' behaviors and then evaluating those behaviors, and a rubric designed based on definitions from various associations [71].

Lastly, for the N.A. (None-applicable) category, all studies have implemented certain assessment methods. However, it is not identifiable whether they are using an existing tool or a tool created by the researchers, such as assessing via interview without providing the rubrics in the article [80].

3.3.4. RQ4: Assessment Contexts

Table 3.5 presents the following aspects of assessment contexts: the educational setting, the academic domain, the educational level, and the nature of learning tasks.

Regarding the educational setting, most of the studies were conducted in a formal setting either in a course or within a curriculum. For example, Libeskind-Hadas and Bush [90] measured CT skills in a computing course with applications to biology, and Mishra and Iyer [72] conducted field studies in the two Computer Science application courses and measured CT. The other studies were categorized as informal/Unspecified, meaning they were neither embedded in a course nor a relevant curriculum. For example, using the Hi-ACT survey, Debby Erce Sondakh et al. [84] conducted a pilot study to assess undergraduates' CT proficiency across domains in different universities. Similarly, Lee and Cho [64] surveyed 159 university students in Korea using questionnaires on CT and students' academic grades without information on a course or curriculum.

As for the academic domain, most of the studies were conducted in a STEM context. Some implemented multidisciplinary or interdisciplinary studies, such as Wang et al. [88], an interdisciplinary course that simultaneously engages the disciplines of engineering, science, and arts. A few studies cover only non-STEM domains; for example, Cabo and Lansiquot [54] integrated CT into liberal arts courses aimed at non-computer majors.

Table 3.4.: Categorization of results on assessment approach (RQ3)

Category	Sub-codes	Reference
Assessment Perspective	Peer assessment (1)	[48]
	Self-assessment (7)	[61] [62] [76] [77] [78] [84] [85]
	Teacher assessment (35)	[50] [45] [51] [52] [53] [54] [56] [57] [58] [59] [60] [47] [63] [46] [64] [65] [66] [67] [90] [68] [70] [71] [72] [73] [74] [49] [79] [80] [81] [82] [83] [91] [86] [87] [88]
	Mixed use (4)	[89] [55] [69] [75]
Assessment Format	Automatic (4)	[51] [53] [46] [49]
	Manual (36)	[50] [52] [89] [48] [54] [55] [56] [58] [59] [60] [61] [62] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [76] [77] [78] [79] [80] [82] [83] [91] [84] [85] [86] [87] [88]
	Both (7)	[45] [57] [47] [63] [90] [75] [81]
Assessment Method	Interview (2)	[47] [80]
	Journal (2)	[45] [88]
	Questionnaire & Survey (9)	[61] [62] [68] [76] [77] [78] [84] [85] [86]
	Traditional test & exam (18)	[50] [51] [52] [53] [55] [59] [46] [64] [67] [68] [69] [70] [72] [73] [77] [82] [83] [91]
	Miscellaneous (29)	[45] [89] [48] [54] [56] [57] [58] [60] [47] [63] [46] [65] [66] [67] [90] [68] [69] [71] [72] [74] [75] [49] [79] [81] [82] [83] [91] [87] [88]
Assessment Tool	Adaptation of an existing tool (5)	[56] [59] [64] [65] [77]
	Direct use of an existing tool (18)	[45] [52] [89] [48] [53] [55] [57] [58] [60] [47] [63] [46] [66] [90] [73] [49] [81] [87]
	Originally invented tool (17)	[51] [62] [63] [90] [68] [69] [70] [71] [74] [75] [76] [78] [81] [83] [84] [85] [86]
	N.A. (None-applicable) (10)	[50] [54] [61] [67] [72] [79] [80] [82] [91] [88]

3

Table 3.5.: Categorization of results on assessment context (RQ4)

Category	Sub-codes	Reference
Educational Setting	Formal (39)	[45] [51] [52] [89] [48] [53] [54] [55] [56] [57] [58] [59] [60] [47] [62] [63] [46] [65] [66] [67] [90] [68] [69] [70] [71] [72] [76] [77] [49] [78] [79] [80] [81] [82] [83] [91] [86] [87] [88]
	Unspecified (8)	[50] [61] [75] [64] [73] [74] [85] [84]
Academic Domain	CS (14)	[50] [45] [52] [89] [59] [61] [46] [69] [72] [75] [77] [78] [80] [87]
	Non-CS (25)	[50] [53] [54] [55] [57] [60] [47] [62] [65] [66] [67] [90] [68] [70] [71] [73] [74] [75] [76] [77] [49] [81] [82] [91] [87]
	N.A. CS/Non-CS (12)	[51] [48] [56] [58] [63] [64] [79] [83] [84] [85] [86] [88]
	STEM (29)	[50] [45] [52] [89] [48] [55] [58] [59] [61] [63] [46] [64] [65] [67] [90] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [49] [78] [79] [80] [82] [91] [84] [85] [86] [87] [88]
	Non-STEM (13)	[53] [54] [57] [60] [64] [74] [75] [76] [77] [81] [84] [85] [88]
	N.A. STEM/Non-STEM (6)	[51] [56] [47] [62] [66] [83]
Educational Level	Graduate (2)	[50] [57]
	Undergraduate (41)	[45] [51] [52] [89] [53] [54] [55] [56] [58] [59] [60] [47] [61] [62] [46] [64] [65] [66] [67] [90] [69] [70] [71] [72] [73] [74] [76] [77] [49] [78] [79] [80] [81] [82] [83] [91] [84] [85] [86] [87] [88]
	Both (4)	[48] [63] [68] [75]
Nature of Learning Task	Programming (3)	[52] [67] [78]
	Non-Programming (5)	[62] [46] [65] [83] [86]
	Both (35)	[50] [89] [48] [54] [55] [56] [57] [58] [60] [47] [63] [66] [90] [68] [69] [71] [72] [76] [77] [49] [79] [82] [91] [87] [88]
	N.A. (14)	[45] [51] [53] [59] [61] [64] [70] [73] [74] [75] [80] [81] [84] [85]

For the educational level, most studies were at the undergraduate level, while two were at the graduate level, and four covered both levels. For example, Alrashidi et al. [50] recruited Masters and PhD students, Libeskind-Hadas and Bush [90] included first-year undergraduate students, Altanis and Retalis [45] included last-year undergraduates, and Basawapatna and Repenning [48] involved undergraduate/graduate level students.

In terms of the nature of the learning tasks, most studies adopted a combination of both programming and non-programming tasks. An example of using both programming and non-programming tasks is a multi-disciplinary study [88] where students learned knowledge from CS, Interactive Multimedia, Mechanical Engineering, and Music to implement an open-ended team project. Only 3 and 5 studies used programming and non-programming tasks for learning, respectively. The studies [14, 78] that were in programming courses and used programming-related tasks for learning and assessing, were considered as the programming category. Meanwhile, studies that adopted programming concepts into other activities instead of using programming tasks for learning were considered non-programming categories; for example, Takemata et al. [86] taught students CT skills by teaching digital storytelling instead of asking students to program code.

3.4. Discussion

As different assessment purposes require different design decisions [94], it is essential to identify the assessment purposes when designing and conducting assessments. The results show a clear focus on summative assessment which functions primarily as providing specific learning performance results to the learners or teachers. However, to understand and assess the level of competence and understanding students' mastery of complex concepts such as CT, the development and trajectory of developing a solution are often much more insightful than the mere snapshot of a final solution [95]. Therefore, we argue that formative assessment and support of learners while developing CT solutions opens many more opportunities for assessment and learning support.

Considering the richness of the assessment, we found a broad range of constructs and concepts assessed, ranging from Computer Science (process and Psychology of coding) to affective and attitudinal perspectives (teacher attitudes, acceptance). Moreover, results show that most assessments are manually conducted in formal settings in STEM domains (especially CS majors) at undergraduate levels, either using an existing tool or creating a new tool according to the needs. While various naming schemes and varied approaches to measure CT in different domains provide the designers with a broad spectrum to choose from, we argue that more integration of different perspectives is needed in different disciplines. What is more, the use of one type of assessment method is insufficient to capture the performance of students given the richness of the constructs available and the complexity of the constructs to assess in different domains. Therefore, we suggest an interdisciplinary approach to CT as we see a focus on mono-subject teacher-driven assessment and also a strong focus on CSE.

3.4.1. Limitations

Though we strove to conduct the research as rigorously as possible, we identified several limitations. The limitation can stem from the database selection (we used only two databases), the determination of the search terms, the search strategies applied in the databases, and the employed inclusion and exclusion criteria. Moreover, recent studies could have been excluded from this analysis due to the final date of the search conducted. Lastly, the coding scheme could also be a limitation of this study.

3.5. Conclusions

This review systematically reviewed empirical studies on CT assessment in higher education to gain insights on aspects to be considered for assessment design, including assessment purposes (why), assessment constructs (what), assessment approaches (how), and assessment contexts (where, who, when). Following a systematic review method, we reviewed 47 empirical studies published between 1986 and 2021. Findings reveal that most assessments were conducted with a summative purpose in STEM fields at the level of undergraduates, assessing CT constructs in various naming and categorization schemes, with mainly existing assessment tools or new ones designed initially by the authors. Our main conclusions from this study are twofold. First, considering the design of assessment methods for CT, we recognize that there is a further need to a) understand the process instead of the final result, indicating the necessity of methods for formative assessment and monitoring of CT problem-solving. Current developments in Learning Analytics support this argument by introducing analytics sequences in problem-solving and code development. Furthermore, we b) recognize that there is a broad range of methods to assess CT from different perspectives to choose from, also established in currently used frameworks. Second, most studies we found take a very mono-disciplinary perspective on CT and do not often integrate indicators from Computer Science (Process and Psychology of Coding), Cognitive Psychology (Knowledge, Skills, and Competencies), or Affective and Attitudinal perspectives (teacher attitudes, acceptance). We would strongly argue for an interdisciplinary approach to link these perspectives in a holistic CT assessment approach.

Part II Computational Thinking Integration in Higher Education in Practice

4

An Investigation on Integration of Computational Thinking into Engineering Curricula at the Delft University of Technology

We do not learn from experience. We learn from reflecting on experience.

John Dewey

Parts of this chapter have been published in SEFI 2022 - 50th Annual Conference of the European Society for Engineering Education, Proceedings, 890–901 [96].

Digital devices surround our lives. Engineering education is essential in higher education for future generations, and computational thinking (CT) is a core component in various engineering curricula. At the Delft University of Technology (TU Delft), computational concepts and activities have been integrated into the curriculum for years. However, a comprehensive investigation of the integration of CT into the Engineering Curriculum is yet to be performed. This study examines Master's level engineering curricula from the following aspects: 1) to what extent CT components are integrated; 2) in what way CT is interpreted, and integrated into the curriculum; 3) what educational and assessment methods have been used. The results show that CT has been largely integrated into the curriculum, mostly applying lectures as the educational method and programming assignments for assessment. Results indicate that understanding the context, recognizing the patterns in problems, and organizing patterns in solutions are essential in various engineering disciplines.

4.1. INTRODUCTION

The nature and shape of engineering education is an ongoing topic of debate among engineering faculty members, professionals, and practicing engineers [97]. It appears that the enormous changes in societal dynamics and the development of technology also lead to the transformation of engineering curricula. In both the United States and Europe, an emphasis has been placed on digital skills and literacy. In 2016, Barack Obama [2] launched the 'Computer Science for All' initiative. This initiative aimed to allow American students from kindergarten through high school to learn computer science and acquire computational thinking (CT) skills. Computer science was referred to as a new basic skill necessary for economic opportunity and social mobility. As for the European Union, the Joint Report of the Council and the Commission on the Implementation of the Strategic Framework for European Cooperation in Education and Training[3] highlights the importance of digital competencies. In addition, the study Developing Computational Thinking in Compulsory Education by the Joint Research Centre, which is the European Commission's Science and Knowledge Service, suggests that CT is a subject of importance within the European Union.

Addressing this emphasis on digital competencies and CT, in the Netherlands, 4TU has been created to foster collaboration between the four universities of technology (TU Delft, Eindhoven University of Technology, University of Twente, and University of Wageningen). In the vision for Higher Engineering Education written by Kamp [2], the 4TU Centre of Engineering Education in the Netherlands is mentioned as a "Free-Spirits" think tank that aims to develop suitable higher education scenarios by 2030 by examining the appearance of new engineering profiles in the coming 10 to 15 years. Inspired by the Greek Philosopher Heraclitus stating "the only constant in life is change", its vision states that digitalization is one of the driving forces which makes our world more uncertain, complex, and ambiguous. This not only impacts how people live but also requires future generations of students to be empowered with lifelong learning and general problem-solving competencies including the use of digital and computational tools.

CT was first mentioned by Papert[98] and then advocated by researchers and practitioners since Wing claimed it was "a must-have skill for everyone" living in the 21st Century rather than solely for computer scientists. Ever since there has been extensive debate and research on the definition of CT and its relevance for education. Several definitions are used in the field, for example, Wing [18] defined CT as a set of problem-solving skills with which the formulation of solutions for problems can be carried out by computing agents (either human or mechanics computing machines). Unlike Wing's definition, which is descriptive and theoretical, some operational definitions attempt to identify more granular constructs that are more practical. Brennan and Resnick's [22] three-dimensional framework consists of CT concepts, CT practices, and CT perspectives as well as the four compositional frameworks with problem decomposition, pattern recognition, abstraction, and algorithm are frequently used in research and practices. While efforts to concretize the definition of CT are still ongoing, the importance of helping people adapt and prepare future generations to live in a digital society is widely recognized. As a result, researchers, educators, policymakers and practitioners from both Science, Technology, Engineering, and Mathematics (STEM) and non-STEM backgrounds are currently investigating the integration of CT into different fields.

Bearing in mind that fostering students' CT skills is a strategic focus of the 4TU, in this work we aim to investigate their integration into engineering curricula. Considering the effort needed for the investigation of all relevant faculties, we limit our case study to the MSc programs of the two faculties of TU Delft that are inclined to integrate CT into their curriculum, and we specifically aim to answer the following research questions:

RQ1 To what extent is CT reflected in the curricula of the TU Delft?

RQ2 In what way does the interpretation or reflection of CT differ per faculty?

RQ3 Which educational and assessment methods are used in CT-integrated courses?]

4.2. Method

4.2.1. Research Scope and Source Data

The scope of this study was limited to the MSc programmes of two TU Delft faculties: Architecture and the Built Environment (ABE), and Electrical Engineering, Mathematics & Computer Science (EEMCS). As our data source, we used the TU Delft study guide, in which we initially performed a keyword-based search and then analyzed the course data to answer the research questions. The search criteria were informed after consultations with faculty members. The TU Delft study guide is the formal collection of all courses and study programs offered by the different faculties of TU Delft. Each course is listed and described in this database with information such as its overview of course design and general course arrangement. A search functionality offers the opportunity to search the course descriptions per faculty for one or multiple keywords. Hence, the study guide was used as the basis for this systematic research into how CT is reflected across the faculties of the TU Delft.

4.2.2. Identification of Relevant Keywords

To identify courses that discussed CT explicitly, the keywords "computational thinking", "digital skills", and "digital competency" were applied. Besides identifying courses that explicitly discussed CT, we were interested in identifying the courses that implicitly included CT. To identify those courses, we leveraged the operationalization of CT devised by BBC (2018), which comprises problem decomposition, pattern recognition, abstraction, and algorithm design. For each of the four components of CT, keywords were defined that signaled the presence of that step of CT (Table 4.1). The asterisk (*) in some keywords ensures courses are found where both words occur in the text, although not sequentially. In addition, it allows for finding both UK and US spelling in e.g. mode*ing. In addition to these general keywords, after consultation with an associate professor for the faculty of Architecture and the Built Environment, the keywords "parametric", "computational", " algorithm", and "simulation" were added as ABE-specific keywords.

Four Components	Keywords
Problem decomposition	Problem*decomposition; Problem*data; Problem*pattern; Problem* abstraction
Pattern Recognition	Pattern* recognition; Data*analysis; Data*creation; Data*collection; Data*representation; Identif* patterns
Abstraction	Abstraction; Generali*ation; General*solution; Formulating* solution
Algorithm Design	Algorithm* design; Algorithmic* thinking; Algorithm* problem; Algorithm* pattern; Algorithm* solution; Computational* problem; Computational* pattern

Table 4.1.: Overview of the keywords used to identify courses that implicitly include CT

4.2.3. Filtering of the Relevant Data

Several filtering steps were applied after performing the keyword search (see Figure 4.1). First, BSc courses were discarded since we were solely interested in MSc courses. Since a considerable number of courses within EEMCS included one of the CT components, only the courses with at least two CT components were evaluated. Then, while the presence of one or more CT components under each component in the course description signaled the possibility of CT, it was found that merely the appearance of those components was not sufficiently specific to signal the presence of CT. Therefore, the keywords and their corresponding CT components were applied, and an evaluation of the course description in the study guide was performed to identify the courses reflecting CT. The courses were scored on the presence or absence of each of the four components of CT. The course descriptions which at least signaled the presence of two out of the four components of CT were included in our validation set while courses with less than two components were discarded. The validation sets were sent to three CT experts (second evaluators) for evaluation together with the operationalization of CT. The courses that were rated by at least three out of four raters as CT were included in the final selection.

4.2.4. Data Extraction and Synthesis

The selected courses were analyzed with the information extracted from CT courses. Types and categories of the assessment method, educational methods, courses, and other categorizations were based on observation from the dataset and aggregated by one of the authors with consultation from experts when necessary. Table 4.2 in the Appendix provides an overview of the extracted information per CT course (with the last row presenting the information aggregated based on descriptions of the courses).

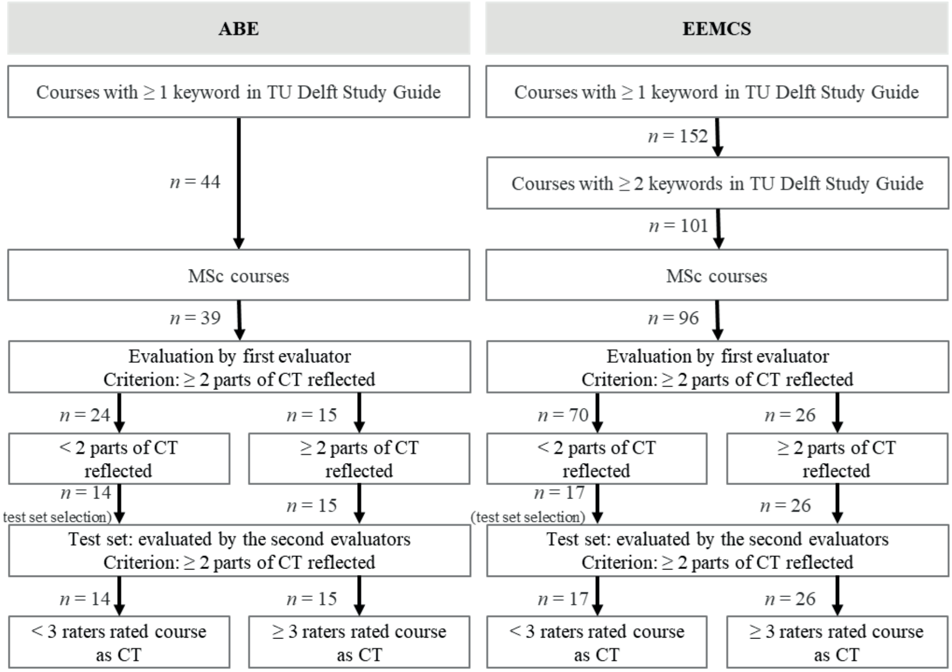


Figure 4.1.: Selection process of the CT courses. ABE: (Architecture and the Built Environment), EEMCS: (Electrical Engineering, Mathematics & Computer Science)

Categories	Sub-categories
General information	Date of consultation; Keyword hits; number of keywords; Course Title; Course code
Place in Curriculum	ECTS; Semester (Q1[1] Q2 Q3 Q4); Programme; Track; MSc year; Elective vs obligatory vs obligatory with choice
Study Description Information	Link; Offered by; E-mail; Course content – TOPIC; Study goals – ALL; Study goals – TOPIC; Education methods; Ass. Type; Ass[2]. Process; Ass. Criteria
Evaluation	Presence of: Problem decomposition, Pattern recognition, Abstraction, Algorithm design; Type of course; Sure (yes/no); Notes

Table 4.2.: Extracted information per CT course

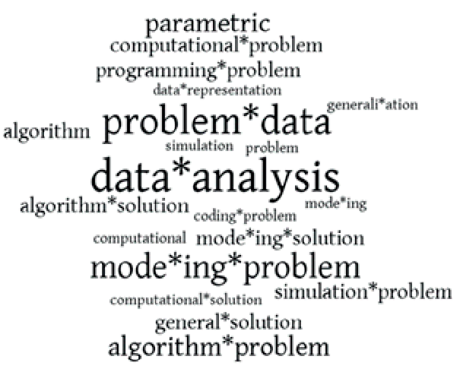
4.3. RESULTS

4.3.1. INTER-RATER RELIABILITY ON THE IDENTIFICATION OF THE RELEVANT COURSE

We found a low Fleiss' kappa inter-rater reliability of 0.33 and 0.17 for the courses of the faculties ABE and EEMCS, respectively. These low values indicate that, although the operationalization of CT of this study was provided to the raters, their interpretations of CT were still divergent. The provided operationalization left room for interpretation, which led each rater to use their specific background knowledge on CT. Zooming in on the ratings, we observe e.g. that one expert utilized a broad perspective and considered almost all courses of the test sets to be CT, while another expert utilized a narrow perspective and only scored about half of these courses to be CT. Their interpretations of CT differ greatly.

4.3.2. CT IN THE CURRICULA

A total of 15 ABE MSc courses under two MSc programs (Geomatics and Architecture, Urbanism and Building Sciences (AU&B)) were designated as CT courses while a total of 27 EEMCS MSc courses were designated as CT courses (an additional filtering step was applied compared to ABE). The keywords that were found in the course descriptions of the CT courses are visualized in Figure 4.2a and Figure 4.2b, with the font size correlating to frequency. Compared to the keywords from ABE, we can see "algorithms" and "algorithm design" were much more frequent for EEMCS. Meanwhile, Of the three MSc programs analyzed, unsurprisingly, Computer Science contained the highest number of CT courses. Due to the set-up of the EEMCS MSc programs, all courses were compulsory choices courses.



(a) ABE

(b) EEMCS

Figure 4.2.: Word cloud of the keywords in the CT courses. A larger font size correlates with a higher occurrence

Figure 4.3 and Figure 4.4 demonstrate the different types of CT courses within ABE and EEMCS respectively. Most courses at ABE were about learning to use a computer program or a modeling task that contributes to analyzing the environment or the design of new buildings. For example, in Digital Terrain Modelling students were taught to use datasets to reconstruct a terrain and use these for applications related to the built environment, and in Geomatics as support for energy applications students are taught to build 3D city models. Another type of CT course at ABE were courses that comprised large projects with a computational component of varying extent. For example, in MEGA students designed a special big building in multi-disciplinary teams. Within this team, just one of the students was responsible for the computational design. On the other hand, in Earthy the design of the building is completely computational. Finally, the course Operations Research Methods was specifically on teaching research methods and using mathematical modeling to make decisions.

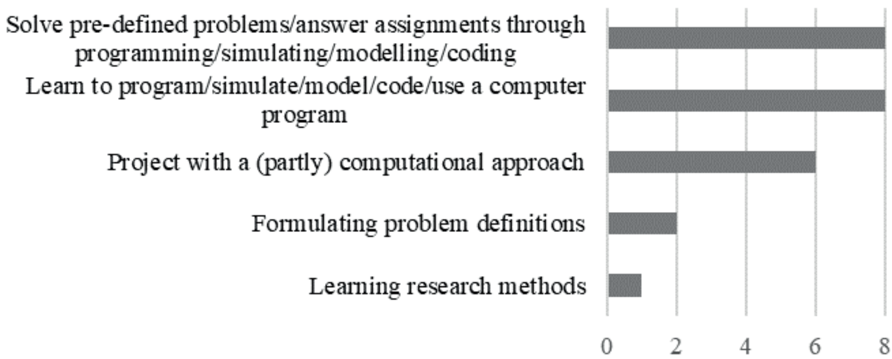


Figure 4.3.: ABE – Types of reflection of CT in courses

As for EEMCS, the largest part of the courses includes learning about the context and patterns in problems and solutions; frequently in the form of students being taught to recognize which algorithms to use for which problems. For example, in Object Classification with Radar two of the four learning goals specifically refer to recognizing patterns in the data and the type of problem and deciding on the approach dependent on these patterns: 'Analyse and compare the different domains of radar data that can be exploited for objects classification, and the convenience of use of one rather than the other in different situations' and 'Propose and evaluate possible approaches for given classification problems based on radar sensors data'. Furthermore, more than half of the courses involved a larger project with a (partly) computational approach. Interestingly, whereas the ABE faculty always started with a problem (e.g., I want to design a building and I need to know the terrain) and then took a computational approach to solve the problem, the EEMS faculty also frequently took the algorithm as a starting point and then went looking for a problem to be solved, sometimes looking at other faculties of the TU Delft to provide problems to be computationally solved.

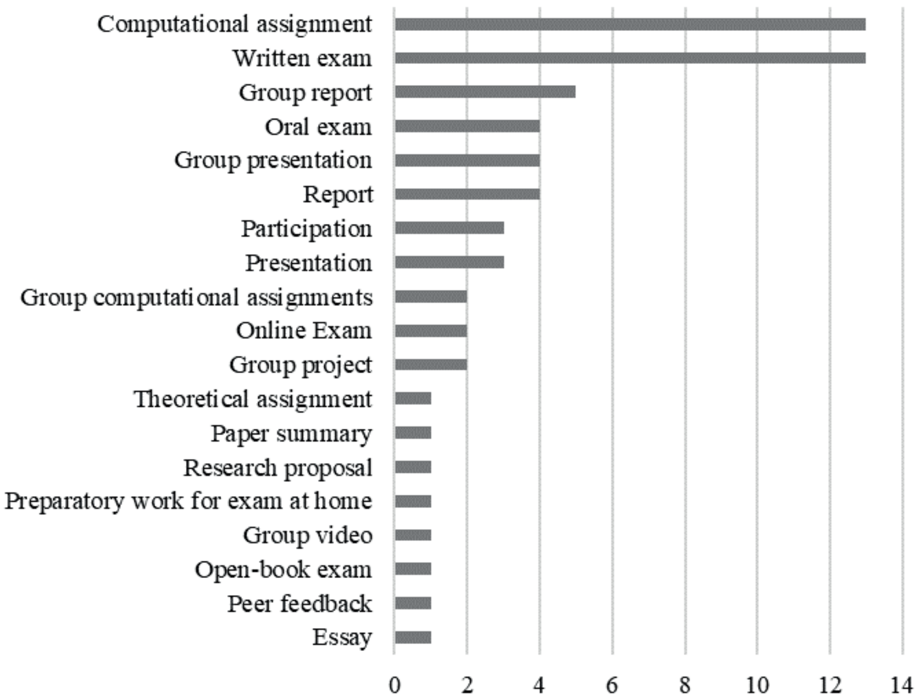


Figure 4.4.: EEMCS - Types of reflection of CT in courses

4.3.3. EDUCATIONAL METHODS

Zooming in on the different educational methods, we see that a wide variety of methods was used in CT courses at ABE (See Figure 4.5). Contrary to what one might expect given the practical nature of CT, lectures were the most common instructional method. However, within one course approximately 4 educational methods were used, signaling that lectures were often used in conjunction with different educational methods. For example, in the course Geomatics as Support for Energy Applications three education methods are used: lectures to provide the theoretical background, practicals to allow the student to practice building models while supervision is present, and self-study to further dive into the theoretical background and do more modeling. Besides lectures, practicals, and self-study, computational assignments were a common educational method, which is in line with expectations: computational assignments allow the student to apply their CT skills in practice. In addition, notable is that a fair number of courses included tutoring sessions. These sessions are a common way of teaching in the ABE faculty, during which a tutor provides feedback on the work or design of the student or students.

Figure 4.6 demonstrates the utilized educational methods at EEMCS, with on

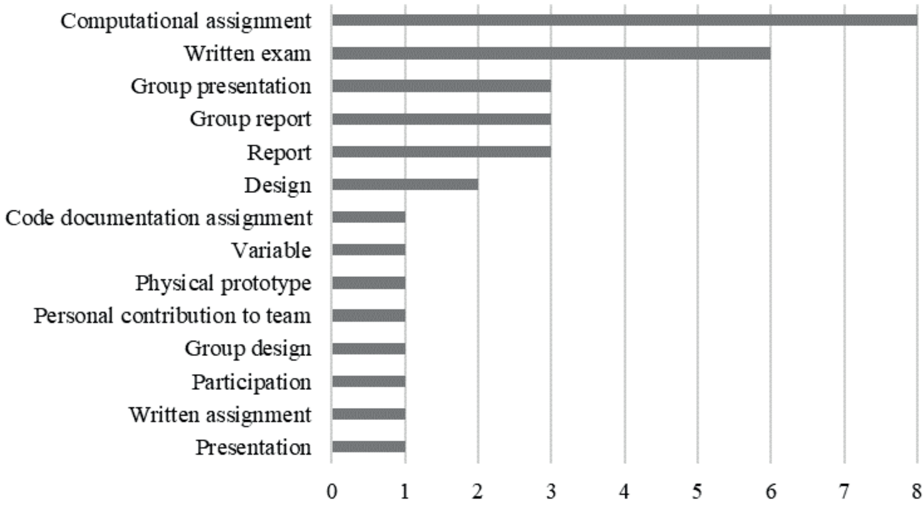


Figure 4.5.: ABE - Occurrences of different educational methods in CT courses

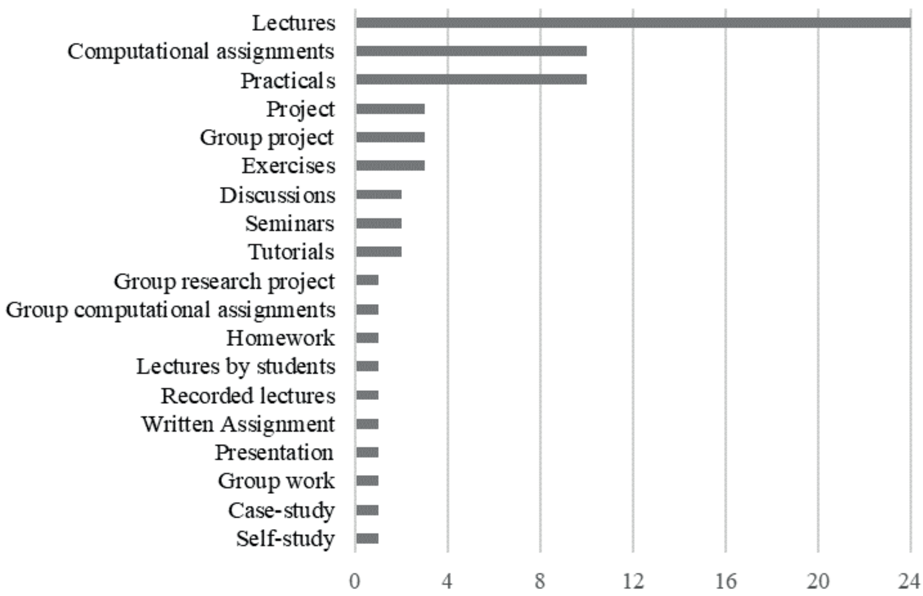


Figure 4.6.: EEMCS - Occurrences of different educational methods in CT courses

average 3 educational methods per course with lectures being the most used educational method. In 37% of the courses, practicals and/or computational assignments were used as educational methods. Larger computational groups or individual projects were more common in the EEMCS faculty. For example, in the course Crowd Computing students are working throughout the entire course for six hours per week in a group on an extensive computational project. Besides this group project, the other utilized education methods are lectures and computational assignments. Notable is the occurrence of some more innovative instructional methods like recorded lectures for the course Applied Machine Learning and lectures by students for the course Machine Learning in Bioinformatics.

4.3.4. ASSESSMENT METHODS

For the assessment at ABE and EEMCS, most courses used more than one assessment method, and an overview of assessment methods being used for both faculties is presented in Figure 4.7 and Figure 4.8 respectively, with the computational assignment (s) being the most common assessment method. The more traditional written exam was also frequently used, often in combination with computational assignments e.g. Geoweb Technology, Geographical Information Systems (GIS), and Cartography and Python Programming for Geomatics. Group presentations, group reports, and individual reports were also common; it should be noted that often these reports were about the developed computational model. For example, in Operations Research Methods the assessment was based on a written assignment and a report on two mathematical models. In addition, they were often used in parallel (e.g., Algorithms for Intelligent Decision Making, Artificial Intelligence Techniques, and Evolutionary Algorithms). Oral exam is also used as an assessment type; though it might be labour-intensive.

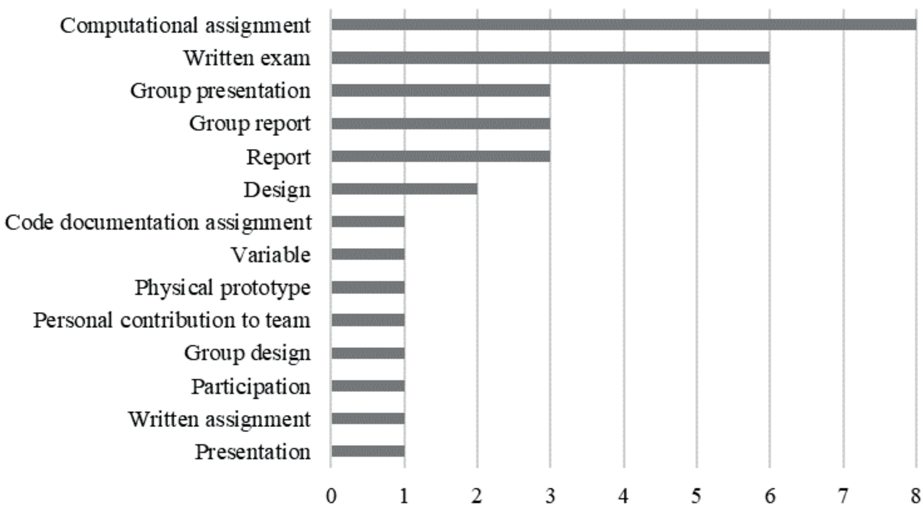


Figure 4.7.: ABE - Occurrences of different assessment methods

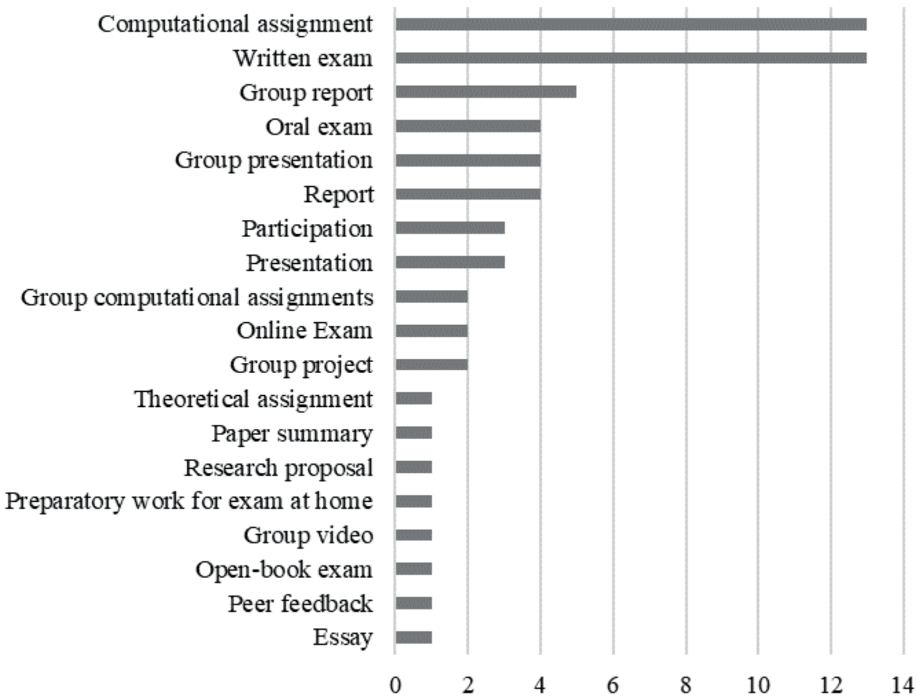


Figure 4.8.: EEMCS - Occurrences of different assessment methods

4.4. Conclusion, Limitations, and Future Work

With the preliminary results of this work so far, several conclusions can be made: First of all, aligned with the findings of the group concept mapping study from Specht et al. (2019), this work finds that agreement on definitions and the relations between different competences and skills under CT is still vague even for experts. For developing an embedding of CT skills in an engineering curriculum necessary definitions and focus should be aligned. Secondly, the examples from the two different faculties of TU Delft show different approaches and embedding CT concepts into the curriculum. On the one hand, fundamental developments or algorithms and computational abstractions need to be developed extending CS curricula and their focus, on the other hand, more design-oriented engineering disciplines embed computational tools and skills often in concrete design and project work. Last, but not least, we observed course designs in which computational tools and digital skills can be linked to specific subtasks of design challenges but also more generic courses that integrate computational tools as a base element of engineering design.

However, the authors spotted major limitations of this work regarding its methodology and its scope, mainly being: that this is a case study on specific faculties of one university and, even though both are large and well-established faculties, they might not be representative of engineering faculties in other countries. Also, the study was based on the information in the study guide which, even if it is a requirement that it is updated yearly, might not fully reflect the instruction methods that the instructors apply. Furthermore, the course descriptions used for this work were limited to the academic year 2020-2021, which indicates that the effectiveness of the findings in this study is time delimited and engineering education is changing with the technological and societal dynamics, which indicates that longitudinal observations and investigations are needed. Regarding the methodology, though this work follows a certain level of systemic and considers the potential bias caused by individual work, several aspects should be noted: Information coding and data synthesis were weak regarding the verification of the validity; the conclusions were made merely with observation and analysis from course descriptions, making it weak as actual situations may differ from the course descriptions.

To further advance this work, the authors plan to conduct focus group studies or interviews to gain more insights about integrating CT into the Engineering curriculum and establish more solid coding schemes and verification standards to improve the validity of the extracted information in future work.

5

Teachers' Intention to Integrate Computational Thinking Skills in Higher Education: A Survey Study in the Netherlands

Any genuine teaching will result, if successful, in someone's knowing how to bring about a better condition of things than existed earlier.

John Dewey

Parts of this chapter have been published in 2024 IEEE Global Engineering Education Conference (EDUCON) [99].

While no official policy or teaching framework has been established on Computational Thinking (CT) education in the Netherlands, there have been various initiatives for integrating CT into the curriculum. In this study, we surveyed Dutch tertiary education teachers' perceptions and intentions. Our survey encompassed: (1) teachers' perceptions of CT, and (2) their intentions to integrate CT into pedagogical activities. 38 teachers completed our questionnaire. Results show that teachers possess an inadequate understanding of the relationship between CT and Computer Science, have limited training experiences in CT, and hold differing opinions on when and which constructs of CT should be integrated. Moreover, exhibited a strong positive correlation between performance expectancy, attitude towards CT, and behavioral intention to implement CT in learning activities. To foster the integration of CT in tertiary education, our findings suggest further development of higher education teacher training programs on CT and its relation to CS and further exploration of how to enhance teachers' performance expectancy and effort expectancy.

5.1. INTRODUCTION

Computational Thinking (CT) has gained recognition as a crucial competence for students. The international educational and research communities have witnessed numerous practical and theoretical initiatives aimed at promoting CT in K-12 education [100]. In this context, teacher training has emerged as a key factor for successfully facilitating CT integration in the curriculum [101, 102]. Scholars such as Cuny [103] and Lye and Koh [104] have highlighted the significance of proper training and support for teachers in integrating CT into their daily teaching activities. According to Yadav et al. [5] beside the fact that there is an active discussion shaped by the Academy's report from The Royal Dutch Academy of Arts and Sciences (an advisory body to the Dutch government) in 2013, as well as explorative studies for example by Thijs et al. in 2014 [6] there is not yet an official national policy on integrating CT in the curriculum. It is worth noting that most existing studies on CT in the Netherlands have focused on the K-12 context [9], while scholars have identified the need for more implementations of CT within higher education [8, 41]. To identify the facilitators of CT education in the Netherlands, it is crucial to understand teachers' perceptions and intentions to integrate CT into their daily teaching activities in higher education.

Technology acceptance models, such as the Technology Acceptance Model (TAM) [105] and the Unified Theory of Acceptance and Use of Technology (UTAUT) [106], are utilized to address users' perception and acceptance of new technologies or technological skills [107]. Several studies have applied these models to investigate teachers' perceptions and attitudes toward specific skills. For instance, Oluwole [108] employed TAM to explore the relationship between Information Literacy Skills and technology acceptance, revealing the influence of Internet knowledge on the perceived security of E-marketing. Additionally, Ling et al. [107] investigated teachers' perceptions of CT in Malaysian primary education using TAM-based questions and found a strong correlation between perceived ease of CT integration and teachers' attitudes toward behavioral intention. Similarly, Fessakis and Prantsoudi [109] surveyed Computer Science (CS) teachers in Greek primary and secondary schools regarding their perceptions, beliefs, and attitudes toward CT, employing the Theory of Reasoned Action (TRA) and TAM. One of their findings indicated that teachers considered secondary education as the most suitable level for CT integration.

In the context of the Netherlands, Bruggink [110] utilized the Theory of Planned Behavior (TPB) to identify factors influencing teachers' intention to implement CT in primary education programs, leading to the finding of the significant influence of subjective norm and perceived control on teachers' intention to integrate CT in teaching programs. Most related to our work is that of Specht and Joosse [111], who developed a questionnaire based on the UTAUT model to investigate teachers' attitudes and acceptance factors toward integrating CT skills in primary and secondary education classrooms. They suggested using the UTAUT model to assess teachers' intention to implement coding skills in the curriculum. To the best of our knowledge, although the UTAUT model explains 70% of the variance in behavioral intention and outperforms previous models [106], it has not been applied in the higher education context.

To contribute evidence for designing effective training programs and providing proper support to teachers for the integration of CT in education, our study aims to investigate teachers' perceptions of CT skills and their intention to integrate CT into teaching and learning practices in the higher education context. To achieve this, we adopted items from the UTAUT model [106] and other existing studies that capture teachers' perceptions and intention on CT integration [107, 109, 111]. The research questions guiding our study are as follows:

RQ1 What are higher education teachers' perceptions of CT skills?

RQ2 What factors influence higher education teachers' intention to integrate CT in teaching and learning practices, and how do these factors relate to each other?

Regarding teachers' perception of CT, we examined their previous training, understanding the relation between CT and CS, scoring on the importance of CT constructs, views on who can teach CT, and what education level would be appropriate for CT integration. Regarding teachers' intention to integrate CT into pedagogical activities, we utilized the adopted UTAUT model and tested seven hypotheses.

The following sections present the background knowledge and theoretical bases for this study, the methodology we employed, the results and findings from a questionnaire-based survey of teachers, and finally, a discussion and conclusion based on the results.

5.2. Background and Related Work

5.2.1. CT, CT Frameworks, and the Hybrid Three-Dimensional CT Framework

Over the past few decades, numerous studies have focused on advancing CT education in various aspects, such as curriculum design, the development of teaching materials and tools, assessment frameworks, and teacher training programs. Wing emphasized the significance of CT, positioning it alongside essential skills such as reading, writing, and counting [1]. Research has shown that CT can enhance higher-order thinking skills and improve problem-solving abilities [112–114].

Integrating CT concepts into classrooms and effectively delivering them requires comprehensive exploration and identifying appropriate support mechanisms to assist teachers in their teaching practices. However, one challenge in this regard is the lack of a universally agreed operational definition of CT.

Throughout the years, various operational definitions of CT have emerged, encompassing key concepts, skills, and examples of its integration in different scenarios. For instance, Wing suggested that CT involves dimensions such as *abstraction, problem decomposition, pattern recognition, algorithmic thinking,* and *logical thinking* [1]. Computer Science Teachers Association (CSTA) and International Society for Technology in Education (ISTE) included nine core concepts and capabilities in CT, including *data manipulation (collection, analysis, representation), problem decomposition, abstraction, algorithms and procedures,* *automation, parallelization,* and *simulation* [115]. Other dimensions have been proposed in other studies (e.g. [29, 100]. However, most operational definitions primarily cover limited dimensions and are predominantly adopted in K-12 contexts.

In the context of higher education, the latest comprehensive framework is a hybrid three-dimensional framework from Lu et al. [7], which is derived from Weintrop et al.'s CT framework for mathematics and science classrooms [116], Grover and Pea's two-dimensional framework [117], and Brennan and Resnick's three-dimensional framework [42]. This hybrid framework encompasses 26 dimensions, shown in Figure 5.1, and it serves as the adopted framework in the present study.

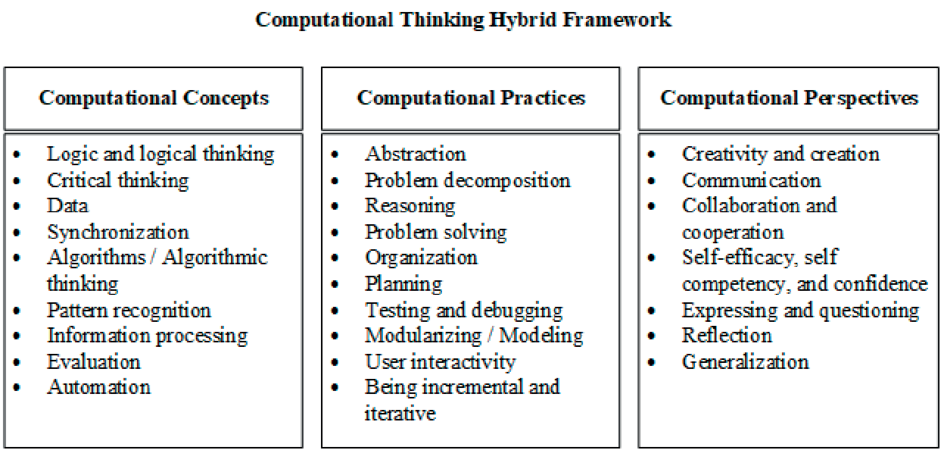


Figure 5.1.: Hybrid three-dimensional CT framework [7]

5.2.2. TECHNOLOGY ACCEPTANCE MODELS

Several theories and models have been developed in the past few decades to examine users' acceptance or rejection of specific technologies or technological skills for decision-making purposes. These models focus on the factors influencing users' intention to implement the technology in their practices.

One well-known model is the TAM, proposed by Davis [105], which originated from the psychological theories of reasoned action and planned behavior. TAM assumes that perceived ease of use and perceived usefulness mediate the relationship between system characteristics (external variables) and potential system usage. Other models that have been utilized in this area include the Theory of Planned Behavior [118], Diffusion of Innovation theory [119], Theory of Reasoned Action [120], Model of PC Utilization [121], Motivational Model [122], Unified Theory of Acceptance and Use of Technology (UTAUT) [106], and Social Cognitive Theory [123].

Several technology acceptance models have been applied to investigate the integration of CT in education. For instance, Ling et al. [107] employed the Technology Acceptance Model (TAM) to explore Malaysian teachers' perceptions of CT in primary education through a questionnaire. Specht and Joosse [111] utilized the UTAUT model to examine teachers' attitudes and acceptance factors regarding the integration of CT skills in primary and secondary education in the Netherlands. Bruggink [110] employed the Theory of Planned Behavior (TPB) to identify factors influencing teachers' intention to implement CT in primary education programs in the Netherlands. Fessakis and Prantsoudi [109] used the Theory of Reasoned Action (TRA) and the Technology Acceptance Model (TAM) to survey Computer Science (CS) teachers in Greece, investigating their perceptions, beliefs, and attitudes towards CT in primary and secondary school.

Among the various models mentioned, the UTAUT model [106], which was developed based on earlier models, has shown superior performance by explaining 70% of the variance in behavioral intention. The UTAUT model comprises four key constructs: effort expectancy (EE), performance expectancy (PE), social influence (SI), and facilitating conditions (FC). Additionally, it includes four moderating variables: age, gender, experience, and voluntariness of use. Specht and Joosse [111] investigated the attitude and acceptance factors by including PE, EE, SI, FC, and attitude (AT) as independent variables. Considering that this study aims to investigate teachers' perceptions of CT skills and their intention to integrate CT into teaching and learning practices, we have adapted Venkatesh et al.'s (2003) [106] UTAUT model through discussions between the authors. The adapted model includes PE, EE, SI, FC, BI (behavioral intention), AT, and voluntariness of use.

5.3. Method

5.3.1. Design

This survey study adheres to the research procedure outlined by Bethlehem [124], encompassing study design, data collection, data editing, non-response correction, data analysis, and publication. The primary research design employed in this study is predominantly quantitative, aiming to investigate the relationship between various factors measured using scale-rated questions. The specific areas under investigation in this research encompass participants' demographic information, their perceptions of CT, and their intention to integrate CT into teaching and learning practices. To examine participants' perceptions of CT, relevant questions have been adapted from existing works, as indicated in Section 6.3.3. For assessing the intention to integrate CT in higher education, in this study, we utilized the UTAUT model for its high explainability [106] similar to how it has been applied in [111]. Within the UTAUT framework, the dependent variable is behavioral intention (BI) to incorporate CT into teaching and learning practices, while the independent variables consist of performance expectancy (PE), effort expectancy (EE), social influence (SI), facilitating conditions (FC), attitude (AT), and voluntariness of use (VU).

5.3.2. Participants - Population and Sampling

The participants of this study consisted of experienced teachers in higher education in the Netherlands, with a particular focus on teachers from non-computer science (CS) domains. To ensure compliance with General Data Protection Regulation

(GDPR) regulations and to facilitate the practicality of conducting survey research, a combination of convenience sampling and referral sampling methods was employed as the sampling approach.

Convenience sampling involved selecting participants who were readily available or easily recruitable for the study. Various strategies were employed, such as displaying QR codes and distributing leaflets during workshops and other institutional events where higher education teachers were likely to participate, as well as utilizing social media platforms like blog posts, LinkedIn, Twitter, and institutional newsletters to disseminate the survey.

Referral sampling, on the other hand, encompassed two techniques. Firstly, network sampling was utilized, wherein a probability sample of a larger population likely to have connections to the target population was obtained by contacting communication officers and different contact persons at institutions such as Delft University of Technology. Secondly, snowball sampling was employed, whereby research participants who volunteered to be part of the study were requested to identify and invite additional individuals who met specific characteristics and were potentially willing to participate in the research. Participants were also encouraged to share the survey study within their network.

Before their participation, the teachers were provided with an informed consent form explaining the purpose of the research. A total of 84 individuals responded to the questionnaire, and out of the total responses received, 38 responses were complete and were deemed suitable for analysis.

5.3.3. MATERIALS - INSTRUMENTATION

A comprehensive questionnaire consisting of a total of 13 questions, including sub-questions within certain items, was employed as the data collection instrument. The construction of the survey involved a systematic process of identifying constructs appropriate for answering the research questions via reviewing relevant literature and engaging in discussions with subject matter experts. The primary author formulated an initial survey draft based on these constructs, which was subsequently refined iteratively through feedback received from both experts and the remaining authors.

The final version of the survey encompassed the following key components: (1) Informed Consent: Participants were provided with an informed consent section outlining the purpose and nature of the study; (2) Demographic Information: Participants were requested to provide demographic details, including gender (Q1), age range (Q2), educational background (Q3), as well as their teaching and training experiences (Q4-Q5); (3) Perceptions and Understanding of CT: The questionnaire included items (Q6-Q10) aimed at assessing participants' perceptions and understanding CT; (4) UTAUT Model Factors: Questions (Q11) pertaining to factors derived from the UTAUT model were incorporated to explore their potential influence on participants' intention to integrate CT into teaching and learning practices; (5) Optional Questions: Two additional questions (Q12-Q13) were provided on an optional basis; one question invited participants to provide supplementary comments on skills they deemed important in higher education within their respective domains; the second question sought to identify whether participants were located outside the Netherlands. An overview of the questions can be found in the Appendix A. The subsequent subsections will elaborate further on the specific items included in the study.

RQ1 - Teachers' Perceptions and Understanding on CT (Q6-Q10)

Questions Q6 to Q10 were specifically included to address RQ1, which focused on exploring participants' perceptions of CT. The selection and adaptation of these questions were guided by relevant scholarly works in the field.

Q6, derived from the study conducted by Ling et al. [107], was incorporated to assess respondents' understanding of computer programming, CT, and computers. This question aimed to gain insights into participants' existing knowledge and comprehension of these concepts.

Q7, adapted from Fessakis [109], served to examine participants' understanding of the relationship between CT and computer science (CS). By utilizing this question, the study sought to gauge participants' awareness of the connection between CT and the broader field of CS.

5

To investigate the significance of various CT dimensions in participants' understanding, Q8 included CT constructs identified in Lu et al.'s [7] comprehensive review. This question aimed to explore participants' perceptions of the importance attributed to different facets of CT.

Additionally, Q9 and Q10, adapted from Fessakis [109], were included to probe participants' views on who should teach CT and when it is deemed appropriate to integrate CT, respectively. These questions aimed to shed light on the perspectives regarding the stakeholders involved in CT education and the optimal timing for its integration into teaching and learning practices.

Last but not least, concerning respondents' perceptions regarding the importance of CT skills, Q12 aimed to explore the skills within the realms of computer science (CS), data science, or machine learning that are deemed relevant to graduates in the respondents' respective domains.

The incorporation of these specific questions aimed to gather comprehensive insights into participants' perceptions of CT and its various dimensions, thereby addressing the research objective outlined in RQ1.

RQ2 - UTAUT Model Structure and Definition of Each construct (Q11)

To address RQ2, we operationalized the constructs and sub-constructs of our research model based on the UTAUT framework proposed by Venkatesh et al. [106]. These constructs were integrated into Question 11 of the questionnaire. This section of the survey enabled the assessment of both the independent and dependent variables outlined in the UTAUT model, utilizing a 5-point scale.

The questionnaire items related to the independent variables were distributed as follows: Performance Expectancy (PE) comprised four questions, Effort Expectancy (EE) included four questions, Attitude (AT) consisted of four questions, Social Influence (SI) encompassed four questions, Facilitating Conditions (FC) involved four questions, and Voluntariness of Use (VU) comprised four questions. Each of these variables was measured using a 5-point scale to capture participants' perceptions.

The dependent variable in the questionnaire pertained to the “intention to integrate CT into teaching" and was assessed through three sub-questions presented in the supplementary file. These questions aimed to gauge participants’ intentions to incorporate CT into their teaching practices and were rated on a 5-point scale.

By employing the UTAUT framework and utilizing a comprehensive set of items, this section of the questionnaire allowed for the measurement of the independent variables and the dependent variable in our research model. The inclusion of these specific items enabled the exploration of participants’ perceptions and intentions regarding the integration of CT into teaching and learning practices.

Hypotheses for the Constructs: Based on the constructs incorporated in the research model, the following hypotheses were formulated:

- **H1**: There exists a positive relationship between performance expectancy and teachers’ intention to integrate CT.
- **H2**: There exists a positive relationship between effort expectancy and teachers’ intention to integrate CT.
- **H3**: There exists a positive relationship between social influence and teachers’ intention to integrate CT.
- **H4**: There exists a positive relationship between facilitating conditions and teachers’ intention to integrate CT.
- **H5**: There exists a positive relationship between attitude and teachers’ intention to integrate CT.
- **H6**: There exists a positive relationship between voluntariness of use and teachers’ intention to integrate CT.
- **H7**: There exists a relationship between different pairs of independent variables.

5

5.3.4. PROCEDURE

SURVEY IMPLEMENTATION AND DISTRIBUTION

In line with the survey design, an online questionnaire was developed using the Qualtrics platform. A copy of the questionnaire can be accessed through a provided link: https://docs.google.com/document/d/1gJcjIzfn0zBcWIXi_vu1im1pK5C2KfhTh8iEux4abLA/edit?usp=sharing. To ensure data anonymization, IP address tracking was disabled in the implemented Qualtrics survey. Before the formal commencement of the survey study, a pilot testing phase was conducted. Professionals in higher education were approached through the authors’ network to seek their feedback and suggestions. This pilot testing aimed to refine the questionnaire and ensure its content validity. Subsequently, the questionnaire was distributed as addressed in Section 6.3.2. The questionnaire yielded a total of 84 responses from October 2022 to May 2023, out of which 38 responses (45.2%) were deemed valid for analysis.

Data Inspection, Editing, Non-response correction

By May 2023, a total of 84 responses were received for the survey. Some of these responses were deemed unusable due to incompleteness or other errors commonly encountered in survey data. To facilitate data processing, including data editing and non-response correction, the responses were downloaded from the Qualtrics platform in Excel files. Incomplete responses and test cases created by the authors were subsequently excluded from the completed sample. As a result, a final sample size of 38 responses remained for further analysis.

5.3.5. Data Analysis Techniques

The collected data underwent several analyses to assess the reliability of the questionnaire, explore descriptive statistics of the sample, and investigate the hypothesized relationships between the measured variables. These analyses included basic data analysis, calculation of Cronbach's alpha coefficient to evaluate questionnaire reliability, and correlation analysis to examine the associations among variables. Additionally, regarding the open-text question Q12, among the 34 respondents who provided answers to this optional open-text question, their responses were analyzed using directed content analysis, applying existing skill categories shown in Figure 5.1.

5.4. Results and Analysis

5.4.1. Descriptive Analysis of the Sample (Q1-Q5 and Q13)

A summary of participant profiles is provided in Table 6.2. According to answers to Q13, all respondents in the study were or had been involved in teaching activities within the Netherlands, with males constituting the majority. The age distribution indicated that most participants fell within the 31-40 age group. Furthermore, a significant proportion of the respondents held a Ph.D. degree and had more than 10 years of teaching experience in the field of Beta domain, as per the categorization outlined in the document provided by the National Research Council of the Netherlands. Additionally, Figure 5.2 presents a word cloud representing the frequency of terms in the responses to Q5. The larger the size of the term, the higher its frequency in the dataset. The most frequently occurring terms in the word cloud include science, design, and engineering.

5.4.2. RQ1: Teachers' Understanding and Perceptions of CT in Higher Education (Q6-Q8, and Q12)

Teachers' training and workshops on CT/CS/Programming were captured with Q6, the results presented in Table 5.2 present the findings. It reveals that 26.3% of the participants reported never attending any workshops or training related to computer science or programming. At the same time, a larger proportion, specifically 63.2%, indicated no attendance in workshops or training specifically focused on CT.

When examining participants' understanding of the relationship between CT and computer science (CS), only 26.3% of the respondents provided the anticipated

Table 5.1.: Descriptive analysis of the sample (Q1-Q5)

Items	**Category**	**Frequency**	**%**
Gender	Female	10	26.3 %
	Male	26	68.4 %
	Prefer not to say	2	5.3 %
Age	21-30	5	13.2 %
	23-40	13	34.2 %
	41-50	10	26.3 %
	51-60	7	18.4 %
	Above 60	3	7.9 %
Education	Bachelor of Science (BSc)	1	2.6 %
	Master of Arts (MA)	1	2.6 %
	Master of Science (MSc)	9	23.7 %
	Ph.D.	25	65.8 %
	Others	2	5.3 %
Experience in teaching in higher education	1-3 year (s)	7	18.4 %
	3-5 year	3	7.9 %
	5-10 year	9	23.7 %
	More than 10 years	19	50.0 %
Teaching Domain Classification	Alfa	6	15.8 %
	Beta	20	52.6 %
	Gamma	5	13.2 %
	Alfa and Beta	1	2.6 %
	Alfa and Gamma	2	5.3 %
	Beta and Gamma	3	2.6 %
	Not applicable	1	7.9 %

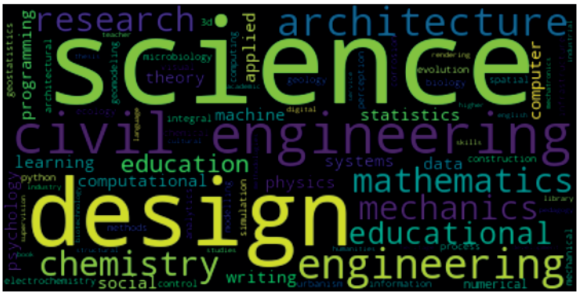


Figure 5.2.: Word cloud for the domains/subjects. (Q5)

response, acknowledging that CT and CS intersect and are not entirely distinct. A majority of the participants (57.9%) expressed that CS is a subset of CT (Table 5.3). These results show that the respondents seem to have an inadequate understanding of the relationship between CT and CS.

Table 5.2.: Training with CS and/or Programming and CT (Q6)

Item	Category	Frequency	%
I have attended workshops / training related to computer science and / or programming.	Never	10	26.3 %
	Once	3	7.9 %
	Sometimes	15	39.5 %
	Repeatedly	6	15.8 %
	Regularly	4	10.5 %
I have attended workshops / training related to Computational Thinking.	Never	24	63.2 %
	Once	2	5.3 %
	Sometimes	7	18.4 %
	Repeatedly	3	7.9 %
	Regularly	2	5.3 %

Table 5.3.: Teachers' understanding of the relation between CT and CS (Q7)

Category	Frequency	%
CT is a concept wider than CS, because it further includes the ability to solve problems in various disciplines, even without the use of computers.	22	57.9 %
CT and CS have common attributes, but each one also has special, discrete attributes.	10	26.3 %
CS is a concept wider than CT, because it further includes, e.g. the study of computation, programming languages, and computer hardware.	6	15.8 %

In terms of respondents' perceptions regarding the importance of CT dimensions and constructs, we examined the mean and standard deviation values on the transformed Likert scale (ranging from 1 for "Not important" to 5 for "Very important"). The findings, as presented in Table 5.4, indicate that all CT dimensions (CT concepts, CT practices, and CT perspectives) are deemed important. Notably, CT perspectives exhibit the highest standard deviation, suggesting greater variability in respondents' perceptions. Furthermore, the correlation analysis conducted on the importance of CT dimensions reveals moderate correlations among all dimensions. This implies that the various CT dimensions are interconnected to some extent. The distribution of the importance values for each CT construct is visually represented in Figure 5.3. It can be observed that most constructs have a median value of 4, indicating a relatively high level of importance attributed to them. However, none of the constructs reached the maximum value of 5, indicating that no respondents considered any of the constructs as "Very important."

Concerning respondents' perceptions regarding the importance of CT skills, Question 12 aimed to explore the skills deemed relevant to graduates in the respondents' respective domains, specifically within the realms of computer science (CS), data science, or machine learning.

By analyzing the answers to the open-text question, we identified eight constructs from CT practices (*abstraction, being incremental and iterative, debugging, modularization/modeling, organizing, planning, problem decomposition, problem-solving*), six constructs from CT concepts (*algorithm / algorithmic thinking, critical thinking, data, evaluation, logic and logical thinking, pattern recognition*), and three constructs from CT perspectives (*collaboration, communication, generalization*).

Among the various responses, the most frequently mentioned CT constructs were *abstraction, algorithm / algorithmic thinking, critical thinking, data, logical thinking,* and *modeling*. Additionally, respondents from different domains often referenced *programming, software engineering, artificial intelligence,* and *machine learning* in their answers, indicating their significance within the context of CT skills.

Table 5.4.: Means and standard deviations for each CT dimension (Q8)

Variable	Mean	Standard deviation
avg_CT concepts	3.47	0.292
avg_CT practices	3.48	0.350
avg_CT perspectives	3.50	0.514

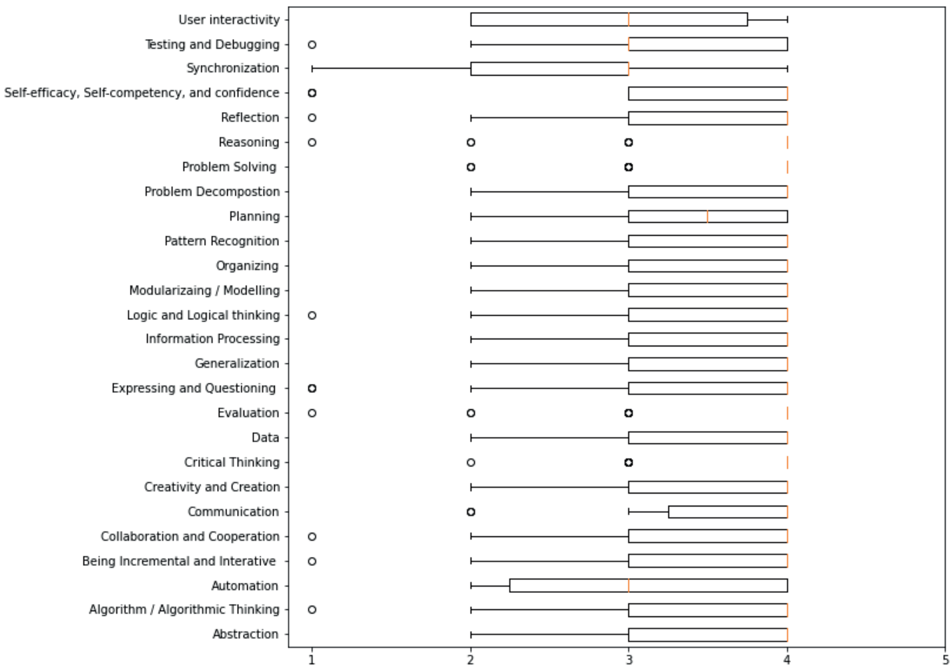


Figure 5.3.: Summary of values on the importance for each CT dimension (Q8)

Question Q9 aimed to investigate respondents' perceptions regarding the individuals who should teach CT. The findings, summarized in Table 5.5, reveal that a majority of the respondents (52.6%) believe that all teachers could effectively teach CT after receiving appropriate training. Conversely, a smaller portion of the participants expressed the view that only teachers with a background in computer science (CS) education should be responsible for teaching CT. Notably, a significant number of respondents hold the belief that anyone can teach CT, irrespective of prior experience in CS.

Regarding the integration of CT into different levels of education (Q10), respondents primarily identified secondary and tertiary education as the most suitable contexts for incorporating CT. In contrast, the responses regarding both primary and secondary education were relatively limited, accounting for only 2.6% of the total, as presented in Table 5.5.

Table 5.5.: Who can teach CT (Q9)

Items	**Count**	**%**
All teachers, regardless of Computer Science (CS) education experience	16	42.1 %
Teachers with Computer Science (CS) education	2	5.3 %
Teachers with proper training in Computer Science (CS) knowledge	20	52.6 %

Table 5.6.: Integration and teaching of CT in education (Q10)

Items	**Count**	**%**
Primary school (primary education)	5	13.2 %
High School (secondary education)	6	15.8 %
Tertiary education (higher education)	4	10.5 %
Primary school (primary education) & High School (secondary education)	1	2.6 %
High School (secondary education) & Tertiary education (higher education)	10	26.3 %
Primary school (primary education) & High School (secondary education) & Tertiary education (higher education)	12	31.6 %

5.4.3. RQ2: Teachers' Intention to Integrate CT - the UTAUT Model

To explore respondents' intention to integrate CT into their teaching and learning practices, we analyzed the responses to the UTAUT items. Firstly, we examined the descriptive statistics of the UTAUT items, assessing their reliability and validity. Subsequently, we calculated coefficients for the variables within the adapted UTAUT model. The mean and standard deviation values for each dimension are presented in Table 5.7, indicating that most dimensions had a mean score above three and a standard deviation higher than 0.6.

To assess the internal consistency of the items, Cronbach's alpha was calculated for each dimension, as shown in Table 5.8. All dimensions demonstrated an acceptable level of internal consistency.

Table 5.7.: Means and standard deviations for each UTAUT dimension used in this study (Q11)

Variable	Mean	Standard deviation
avg_PE	3.51	0.604
avg_EE	3.02	0.796
avg_AT	3.66	0.791
avg_SI	2.43	0.644
avg_FC	2.74	0.744
avg_VU	3.50	0.828
avg_BI	3.20	0.951

Table 5.8.: Construct reliability for each dimension

Construct	Cronbach's Alpha
Performance Expectancy (PE)	0.702
Effort Expectancy (EE)	0.825
Social Influence (SI)	0.924
Facilitating Conditions (FC)	0.715
Attitude (AT)	0.678
Voluntariness of Use (VU)	0.875
Behavior Intention (BI)	0.945

To test each of the hypotheses, Pearson correlation analysis was employed. The correlation model is illustrated in Figure 5.4. Out of the six hypotheses (H1 to H6), five were accepted, as they exhibited strong correlations. Performance expectancy (PE), effort expectancy (EE), social influence (SI), facilitating conditions (FC), and attitude (AT) significantly influenced respondents' behavioral intention (BI) at varying levels of significance, thereby supporting H1 to H5. Among these factors, social influence (SI) demonstrated a relatively weaker correlation with behavioral intention. Notably, there was a slightly negative correlation between respondents' behavioral intention (BI) and voluntariness of use (VU), thus rejecting H6.

Regarding H7, the correlations varied between different pairs of independent variables. Among all the determinants of behavioral intention (BI), attitude (AT) emerged as the most influential factor, primarily influenced by performance expectancy (PE) and effort expectancy (EE).

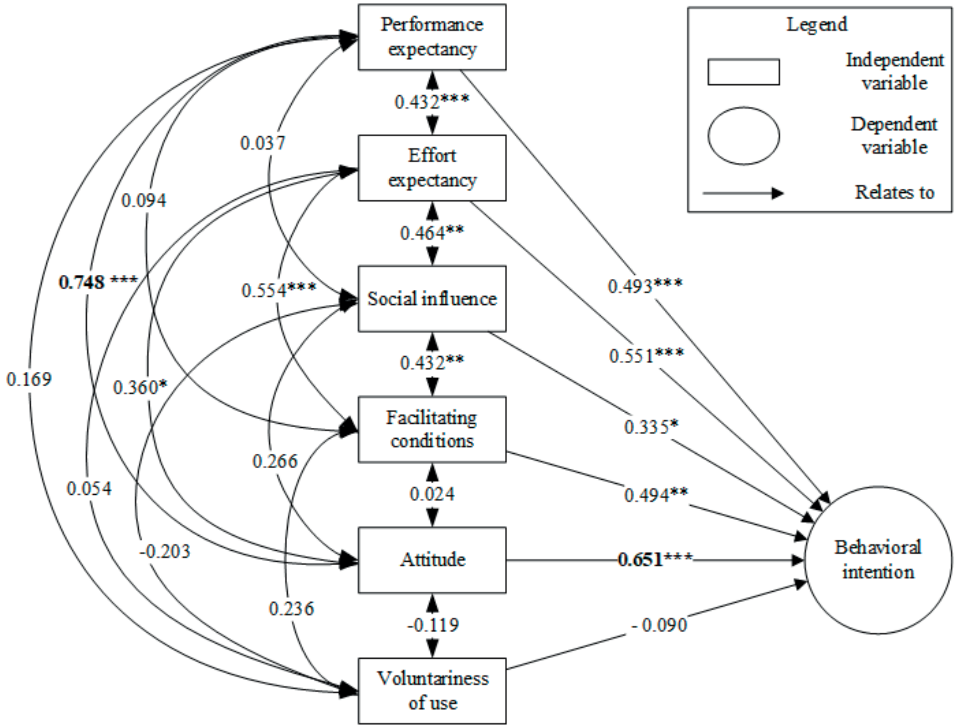


Figure 5.4.: Adapted UTAUT model with correlation values (Q11), *Note. The p values are indicated by the number of *(s): * p <.05, ** p <.01, *** p <.001*

5.5. Discussion

To examine teachers' perception of CT and their intention to integrate CT into their pedagogical activities, we adopted and adapted questions from existing studies, including items adopted from the UTAUT model. Regarding teachers' perception of CT, we examined their training with CS and/or programming and CT, understanding the relation between CT and CS, scoring on the importance of CT constructs, views on who can teach CT, and what education level would be appropriate for CT integration. Regarding teachers' intention to integrate CT into pedagogical activities, we utilized the adopted UTAUT model and tested seven hypotheses, six of which were supported. The findings of this study are further discussed below.

5.5.1. Teachers' perception on CT

As shown in the last section, current CT education remains largely dominated by STEM subjects. A significant proportion of respondents reported never having attended workshops or training related to CT, and a majority demonstrated a low level of understanding regarding the relationship between CS and CT. We speculate that the low understanding level regarding the relationship between CS and CT could be attributed to the lack of CT-specific workshops or training, despite having attended workshops or training in CS. Prior works [125, 126] have emphasized the importance of adequate preparation for teachers to effectively incorporate CT into their teaching and learning practices. This suggests that one potential approach to enhancing CT education is to equip teachers with CT knowledge via structural training. Meanwhile, it is notable that more than a third of the respondents believed that individuals could teach CT regardless of their experience in CS education. The authors advocate here that it is vital to understand the connotation of CT for the teachers, investigate the common ground among teachers, and identify what can be taught by teachers regardless of their experience in CS education. Additionally, the results show that respondents consider all levels of education, from primary to tertiary, as appropriate for implementing CT. This raises questions regarding what should be taught at each level and how to facilitate a smooth transition in teaching practices between different educational levels, if necessary.

The investigation also reveals that six constructs from the hybrid CT framework used in this study (i.e., abstraction, algorithm/algorithmic thinking, critical thinking, data, logical thinking, and modeling) are identified as important for graduates across various domains. These constructs align with those commonly found in K-12 education frameworks, suggesting consistency in curriculum design from K-12 to higher education. However, further exploration is needed to determine how these constructs should be operationalized within the context of higher education. Additionally, respondents identified other skills, such as programming, software engineering, artificial intelligence, and machine learning, as important for their graduates.

5.5.2. TEACHERS' INTENTION TO CT INTEGRATION

The second part of this research aimed to examine teachers' intention to integrate CT skills into their pedagogical activities. All hypotheses, except for H6, received support. The study found that attitude, effort expectancy, facilitating conditions, performance expectancy, and social influence significantly influenced teachers' behavioral intention to integrate CT into their pedagogical activities. Notably, attitude (AT) exerted the most influence on teachers' behavioral intention, with performance expectancy (PE) and effort expectancy (EE) playing major roles in shaping attitudes. Although the original UTAUT model does not emphasize attitude, this finding aligns with studies by Specht and Joosse [111], Ling et al. [107], and Fessakis et al. [109], highlighting the significance of attitude in the integration of CT into education across different levels. Regarding PE and EE, when teachers have higher PE and EE, believing that integrating CT can help them teach well, such as improving their teaching efficiency and quality, and believing that integrating CT is easy, the intention of integrating CT is likely to be higher. This shows that helping the teachers understand the usefulness of CT integration and making it accessible for teachers to implement CT integration is vital for promoting CT integration in education.

5.6. Limitations & Threats to Validity

A threat to the external validity of this study concerns the sample of respondents. Since the participation in the survey was voluntary, the sample can be considered occasional and there was no prior mechanism to ensure its representativeness, which is susceptible to self-selection bias. Even though the respondents reported specializing in a wide range of scientific domains, some disciplines were not represented adequately or at all, which did not allow for exploring interdisciplinary differences. Moreover, it is vital to note that the responses were obtained in the context of higher education in the Netherlands, which might not guarantee the generalizability of results in other contexts.

Concerning construct and internal validity, most of the questions used in this study were adopted from existing studies, the survey was reviewed by all authors and experts, and tested by a pilot study to examine ambiguities. Nonetheless, some items in the survey can still be misinterpreted by participants as the authors and the group of participants involved in the pilot study may not be sufficiently representative of the covered respondents.

5

Additionally, it should be noted that, limited by the method, this research could not explain causality.

5.7. Conclusion

This study investigated higher education teachers' perceptions of CT and their intention to integrate CT via a survey that collected data from teachers in various disciplines. Through this study, we found a necessity to improve teachers' understanding of CT, potentially by offering training and workshops. Moreover, we identified the need to raise teachers' intention to integrate CT into their pedagogical activities within the context of the Netherlands, mainly regarding changing their attitudes that can be dominantly determined by performance expectancy and effort expectancy. Last but not least, discussions on what and when to integrate CT is needed as various views were identified in this study. It should be noted that, while disciplinary differences can significantly influence teachers' perceptions of CT and their intention to integrate CT into pedagogical activities, this study did not allow for exploration in this direction. Therefore, future work can further explore the disciplinary differences with more focused groups of participants.

5.8. Appendix

Table 5.9.: All questions in the questionnaire

Category	No.	Question Type	Question
Gender	Q1	MCQ[1]	I am a: A. Male B. Female C. Non-binary / third gender D. Prefer not to say

[1] Multiple-choice question

Age Range	Q2	MCQ	My age is in the range of: A. 20 or below 20 B. 21-30 C. 31-40 D. 41-50 E. 51-60 F. Above 60
Educational background	Q3	MCQ	My highest academic degree is: A. Bachelor of Science (BSc) B. Bachelor of Arts (BA) C. Master of Science (MSc) D. Master of Arts (MA) E. PhD F. Others ____
Experiences	Q4	MCQ	I have been teaching in higher education (including university, college) for: A. 1-3 year (s) B. 3-5 years C. 5-10 years D. More than 10 years
	Q5	Open text	I have mainly taught topics/areas/domains/majors of (in higher education) ... (If there is more than one, please list them and use a comma to separate different items, e.g. art, civil engineering, statistics, finance.)
Perceptions on CT	Q6	Five-point Likert scale (for two sub-dimensions)	The following statements match my experience with Computer Science (CS) and/or Programming and Computational Thinking (CT): *(1)* I have attended workshops/training related to computer science and/or programming. *(2)* I have attended workshops/training related to Computational Thinking.
	Q7	MCQ	In my understanding, the relation between Computational Thinking (CT) and Computer Science (CS) is: A. CT is a concept wider than CS, because it further includes the ability to solve problems in various disciplines, even without the use of computers B. CT and CS have common attributes, but each one also has special, discrete attributes C. CS is a concept wider than CT because it further includes, e.g. the study of computation, programming languages, and computer hardware D. CS and CT are the same

Q8	Five-point Likert scale (26 sub-dimensions)	I consider the following Computational Thinking (CT) dimensions, in my domain(s) as: (26 dimensions of CT presented for the participants to rank the importance of the construct) • CT concepts: Logic and Logical thinking, Critical Thinking, Data, Synchronization, Algorithm / Algorithmic Thinking, Pattern Recognition, Information Processing, Evaluation, Automation • CT practices: Abstraction, Problem Decomposition, Reasoning, Problem-Solving, Organizing, Planning, Testing and Debugging, Modularizing / Modelling, User interactivity, Being incremental and iterative • CT perspectives: Creativity and creation, Communication, Collaboration and cooperation, Self-efficacy, Self-competency, and confidence, Expressing and questioning, Reflection, Generalization
Q9	MCQ	In my opinion, Computational Thinking (CT) can be taught by: A. Teachers with Computer Science (CS) education B. Teachers with proper training in Computer Science (CS) knowledge C. All teachers, regardless of Computer Science (CS) education experience
Q10	MCQ	I think it is proper to integrate CT into education at (it is possible to choose more than one answer): A. Tertiary education / Higher education B. High school / Secondary education C. Primary school / Primary education

UTAUT model factors	Q11	Five-point Likert scale (27 sub-dimensions)	• **PE** 1. I would find integrating CT into teaching useful in my job 2. Integrating CT into teaching enables me to accomplish tasks more quickly 3. Integrating CT into teaching increases my productivity 4. Integrating CT in teaching increases my chances of getting a raise in salary/employment opportunities • **EE** 5. The way of integrating CT into teaching would be clear and understandable 6. It would be easy for me to become skillful at integrating CT into teaching 7. I would find integrating CT into teaching easy to implement 8. Learning to integrate CT into teaching is easy for me • **AT** 9. Integrating CT into teaching is a good idea 10. Integrating CT into teaching makes work more interesting 11. Integrating CT into teaching is fun 12. I would like to do teaching where CT is integrated • **SI** 13. People who influence my behavior think that I should integrate CT into teaching 14. People who are important to me think that I should integrate CT into teaching 15. The senior management of this school/department has been helpful in the integration of CT into teaching 16. In general, the organization has supported the integration of CT into teaching • **FC** 17. I have the knowledge necessary to integrate CT into teaching 18. I have the resources necessary to integrate CT into teaching 19. Integrating CT into teaching is conflicting with other educational plans of the organization 20. A specific person (or group) is available for assistance with difficulties of integrating CT into teaching • **VU** 21. Although it might be helpful, integrating CT into teaching would certainly not be compulsory in my job 22. My boss would not require me to integrate CT into teaching 23. My superiors would not expect me to integrate CT into teaching 24. Integrating CT into teaching would be voluntary (as opposed to required by superiors/job) • **BI** 25. I intend to integrate CT into teaching in the near future 26. I predict I would integrate CT into teaching in the near future 27. I plan to integrate CT into teaching in the near future

5

Optional questions	Q12	Open text	Which skills related to computer science, data science, or machine learning do you think are important for the graduates in your domain/area/topic?
	Q13	Open text	If you are currently teaching outside of country X, can you please indicate which country that is?

Part III Computational Thinking Assessment in Higher Education in Practice

6

ASSESSMENT OF ALGORITHMIC ABSTRACTION SKILLS IN HIGHER EDUCATION

The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise.

Edsger Dijkstra

Parts of this chapter have been accepted by 2025 IEEE Global Engineering Education Conference (EDUCON)

Abstraction is essential in computer science (CS) and computational thinking education. In the literature, abstraction has been defined on four cognitive levels on which learners can grasp the idea of an algorithm: problem, algorithm, program, and execution level. We designed a study based on these cognitive abstraction levels to examine students' abstraction skills in understanding sorting algorithms. The study investigated the following aspects: (1) the performance of students from CS and non-CS domains on algorithmic abstraction tasks, (2) the association of demographic factors, training experiences, self-perception of proficiency with programming languages, and self-perception of mastery and necessity of algorithmic abstraction skills, with one's performance on algorithmic abstraction tasks. By applying analyses on both qualitative and quantitative data, we found that both CS and non-CS students need abstraction skills at least at the Algorithm Level, with many requiring at the Program Level or the Execution Level. This is especially important for STEM students, highlighting the need for tailored educational approaches for non-CS participants, who show a larger gap between perceived and needed mastery levels. Training and programming experience strongly correlate with performance, emphasizing the importance of targeted training in computational thinking, programming, and algorithmic abstraction. Such training should vary across disciplines, with a minimum requirement at the Algorithm Level. Future work can focus on developing discipline-specific curricula, comprehensive training programs, longitudinal studies on skill development, and cross-disciplinary collaboration.

6.1. INTRODUCTION

Algorithmic problem-solving is crucial for creating computational solutions and promoting computational thinking (CT) in Computer Science (CS) and non-CS fields. It requires diverse skills and knowledge, including basic mathematics, problem-solving techniques, and programming skills. Abstraction is a key element of CT, essential for understanding and solving problems, and translating them into algorithmic solutions [127].

In algorithmic problem-solving, abstraction involves simplifying the problem by removing irrelevant details while keeping its core elements. This process includes refining the solution at different levels until the algorithm is clear and detailed and then converting it into a program. According to Nakar et al.[128], algorithmic problem-solving consists of reformulating the problem, designing an algorithm, implementing it in a programming language, and testing the solution. This method uses a specific form of abstraction known as algorithmic abstraction.

Some scholars believe Nakar et al.'s method [128] may promote computational thinking (CT), but courses often cover only part of algorithmic problem-solving [127]. The relevance of different aspects of algorithmic problem-solving for higher education students remains unexplored. While endeavors are made to promote CT in both CS and non-CS contexts, within which abstraction is essential, most assessments of abstraction skills in higher education focus on self-evaluation and programming tasks in CS. The PGK framework describes four levels of algorithmic abstraction: problem, algorithm, program, and execution levels [16]. By using this framework, Aharoni [129] found that undergraduate students struggled with abstract data structure; Perrenet et al. [15] validated a tool for measuring thinking levels with algorithm questions. The use of the PGK framework to assess algorithmic abstraction skills for both CS and non-CS students in higher education is yet to be explored.

In this study, we designed an instrument by operationalizing the PGK framework to assess algorithmic abstraction skills for both CS and non-CS students. The algorithmic concept adopted in the design is sorting, which is both a basic CS concept and a common concept in our daily lives. The research questions we aim to answer are:

RQ1 What are the differences in performance on algorithmic abstraction tasks for CS and non-CS students?

RQ2 Which factors, among demographic factors, training experiences on CS, programming or algorithmic problem-solving, self-perception of proficiency with programming languages, and self-perception of mastery and necessity of algorithmic abstraction skills, are associated with one's performance on algorithmic abstraction tasks?

This study is organized as follows: Section 6.2 presents relevant background and related research results. Section 6.3 describes the method used for the study, after which results and analysis will be presented in Section 6.4. We provide concluding discussions on the results in Section 6.5.

6.2. Related Work

Studies on CT across different contexts and age levels show that students often struggle with understanding and using abstraction, especially algorithmic abstraction [130]. Methods for teaching algorithmic abstraction vary: some courses use abstraction implicitly without emphasizing the use of it, while others explicitly emphasize the use of it. Findings suggest that explicitly teaching abstraction is more effective for meaningful learning.

Many studies assess abstraction and algorithmic abstraction. For instance, Cutts et al. [130] developed “the abstraction transition taxonomy" to evaluate the degree of abstraction present in tasks, with three levels of abstraction: English, CS speak, and Code. Moreover, Hazzan’s theoretical framework [131] on abstraction levels has demonstrated significant relevance for studies addressing abstraction in CS [132, 133]. According to Hazzan [131], students initially perceive new concepts at a lower level of abstraction to reduce cognitive effort. While this may be a temporary learning stage, some students might remain at this lower level, limiting their understanding.

Building on Hazzan’s work [131], Perrenet et al.[16] developed the PGK hierarchy to evaluate undergraduate students’ understanding of algorithms. The PGK hierarchy has four levels that correspond to different cognitive abstraction stages necessary for a comprehensive understanding of an algorithm. Each higher level is an abstract representation of the one below it, and each lower level is a concrete representation of the one above it. The levels are:

6

- Level 4: *Problem Level* - This is the highest level, where one can reason about an algorithm in terms of the problem it solves and its characteristics. At this level, an algorithm is perceived as a black box, requiring a high degree of abstraction.

- Level 3: *Algorithm (Object) Level* - Here, an algorithm is seen as an object independent of any specific programming language. Complexity measures such as time and space efficiency become relevant at this stage.

- Level 2: *Program (Process) Level* - At this level, an algorithm is viewed as a process, described by a specific executable program written in a particular programming language.

- Level 1: *Execution Level* - The lowest level, where an algorithm is interpreted as a specific run on specific input on a concrete machine.

Following Perrenet et al.[16], many studies have used the PGK hierarchy to examine high school and middle school students’ understanding of algorithmic abstraction [127, 134]. For example, it has been applied in primary school curricula, especially with teachers, where the *Algorithm Level* is seen as a preliminary design phase [127]. In higher education, Aharoni used the PGK hierarchy to study undergraduate students in a data structure course and found that they generally understand data structures at the *Algorithm Level* [129].

6.3. METHOD

6.3.1. STUDY DESIGN

This cross-sectional study followed Bethlehem's procedure [124]. We designed the material to collect participants' demographic information, training experience in programming, CS, and algorithmic problem solving, self-perception in programming language proficiency, self-perception on mastery and necessity of algorithmic abstraction skills, and performance on algorithmic abstraction tasks. Items were revised iteratively based on expert feedback and trials with participants. Data were analyzed using basic descriptive statistics and qualitative analysis.

6.3.2. PARTICIPANTS - POPULATION AND SAMPLING

The study involved Bachelor, Master, and PhD students across diverse disciplines in higher education in the Netherlands. To comply with GDPR and facilitate practical survey research, we applied non-probabilistic sampling methods like convenience and referral sampling. Approval was obtained from the Human Research Ethics Committee of Delft University of Technology. Convenience sampling targeted easily accessible participants through QR codes, online and physical leaflets, social media, and the university's recruitment system. Snowball sampling involved participants inviting others to join the study. Before participating, individuals received an informed consent form detailing the research's purpose. A total of 30 respondents completed the questionnaire and received a 10 Euro compensation voucher upon survey completion.

6.3.3. MATERIALS - INSTRUMENTATION

The experiment material is a questionnaire implemented on the Qualtrics platform, consisting of: (1) questions on participants' demographic information, experience in programming, CS experience, and algorithmic problem solving, self-perception on programming languages proficiency; (2) problem solving tasks based on the PGK framework for the assessment of participants' algorithmic abstraction skills in solving everyday problems; and (3) self-perception on their mastery and the importance of algorithmic abstraction skills. The content is available in Appendix B.

To begin with, for (1), participants were requested to provide demographic details, including gender (Q1), age range (Q2), level of the study program (Q3), year of study (Q4), study domain (Q5-Q6), past training experience in Computer Science, programming, Computational Thinking, or algorithmic problem-solving (Q7), as well as proficiency with programming languages (Q8). The questions asked in this part are mostly multiple-choice questions and Likert scales.

For (2), participants completed tasks based on the PGK hierarchy. Tasks at the *Problem Level* (SP1-Q1 to SP1-Q3) tested their ability to recognize algorithms by the problems they solve, define algorithms, and apply them in different scenarios. Moving to the *Algorithm Level* (SP2-Q1 to SP2-Q6), tasks evaluated participants' skills in creating step-by-step procedures for solutions, considering various versions of procedures and the complexity of the solution, and expressing them in pseudo-code or natural language. At the *Program Level* (SP3-Q1 to

SP3-Q2), participants rearranged disorderly code implementations of algorithms and implemented solutions using Python. Finally, tasks at the *Execution Level* (SP4-Q1 to SP4-Q3) focused on analyzing implementations' execution, debugging issues, and providing strategies for verifying implementation functionality. In this part, almost all questions are open-text questions, a few are multiple-choice and re-ordering questions.

For (3), in two multiple-choice questions (from P-Q1 to P-Q2), the participant was asked to reflect on the necessity of the level of algorithmic abstraction skills and their perceived mastery level. This part only contains multiple-choice questions.

6.3.4. Procedure

Data collection

We created an online environment on the Qualtrics platform with basic multiple-choice and rating questions, along with tasks covering the four levels of algorithmic abstraction as described earlier. Participants were assigned anonymous IDs unrelated to their backgrounds to ensure confidentiality. Before starting, participants received guidance on using the platform and seeking assistance. During the survey, a researcher remained nearby to offer support as needed. Each session lasted approximately one hour.

The experiment yielded 30 records in the dataset. After reviewing the dataset, all valid data underwent editing for further analysis. This editing process involved data pre-processing steps such as cleaning, transformation, coding, and aggregation.

6.3.5. Data Analysis

To understand participants' demographics, training experience in CS or algorithmic problem-solving, programming language proficiency, and perceptions on mastery and necessity of algorithmic abstraction skills, we first applied descriptive analysis to questions Q1 to Q8. These questions are in multiple-choice and Likert scale formats.

To answer RQ1, we scored participants' algorithmic abstraction skills using a rubric[1] based on the PGK framework. Since each level had different total scores, we used Min-Max normalization to standardize the data, transforming the minimum value to 0, the maximum to 1, and all other values to decimals between 0 and 1. We then used an independent student's t-test on the normalized scores to check for significant differences between CS and non-CS participants.

We analyzed tasks at different levels of algorithmic abstraction using thematic analysis based on Kiger et al.'s [135] guidance. Initial codes were generated from the rubric criteria used for scoring tasks. These codes were then organized into themes for each level of algorithmic abstraction as defined by the PGK framework. For the *Problem Level*, we analyzed SP1-Q2 and SP1-Q3, where participants defined the concept of sorting and gave examples of its daily or professional applications. For the *Algorithm Level*, we analyzed SP2-Q3, SP2-Q2, and SP2-Q4. Participants reasoned the applicability of an algorithm, considered its complexity and provided a

[1] The rubric can be accessed via author requests.

step-by-step plan. For the *Program Level*, we analyzed SP3-Q2, where participants implemented their proposed algorithm and reflected on implementation problems. For the *Execution Level*, we analyzed SP4-Q3, where participants described their debugging strategies for a piece of buggy code. The final hierarchy is shown in Table 6.1.

Table 6.1.: Thematic analysis Hierarchy for Qualitative Data from Each Level of Algorithmic Abstraction Tasks

Dimensions	**Themes**	**Codes**
Problem Level	Different understanding of an algorithm concept *(Answers from SP1-Q2 and SP1-Q3)*	Sorting as grouping definition as well as the application of sorting in daily and professional life.
		Sorting as ordering definition as well as the application of sorting in daily and professional life.
		Sorting as both grouping and ordering.
Algorithm Level	Different levels of consideration over the complexity of the algorithm *(Answers from SP2-Q3)*	Consideration over the complexity of the problem.
		Consideration of the strategy for solving the problem.
	Levels of concreteness on the step-by-step plan for the solution *(Answers from SP2-Q2 and SP2-Q4)*	Generic plan on a strategy that does not contain specific operational steps.
		Concrete plan that includes the basic operation of the algorithm.
Program Level	Problems with Implementation *(Answers from SP3-Q2)*	Time, Indication of using existing functions, Python proficiency
Execution Level	Level of specificity on debugging strategies *Answers from SP4-Q3*	Detailed plan for debugging and execution.
		Generic descriptions of overall strategy.

To answer RQ2, we first performed a basic statistical analysis of the multiple-choice answers (P-Q1 and P-Q2) to study their perceptions of the mastery and necessity of algorithmic abstraction skills. We then conducted a correlational analysis to examine how demographic factors (Q1-Q6), programming language proficiency and training experience in CS or algorithmic problem-solving (Q7-Q8), and perceptions of the mastery and necessity of algorithmic abstraction skills (P-Q1 and P-Q2) are associated with participants' scores on algorithmic abstraction tasks.

6.4. Results and Analysis

Table 6.2.: Descriptive analysis of the sample (Q1-Q7)

Items	**Category**	**Frequency**	**%**
Gender	Female	17	56.7 %
	Male	11	36.7 %
	Non-binary/Third gender	1	3.3 %
	Prefer not to say	1	3.3 %
Age	20 or below 20	2	6.7 %
	21-30	25	83.3 %
	31-40	2	6.7 %
	41-50	1	3.3 %
Education Level	Bachelor of Science (BSc)	6	20.0 %
	Master of Science (MSc)	18	60.0 %
	Ph.D.	6	20.0 %
Year of Study	1st Year	9	30.0 %
	2nd Year	8	26.7 %
	3rd Year	6	20.0 %
	4th Year	5	16.7 %
	8th Year	1	3.3 %
	15th Year	1	3.3 %
Study Domain	Non-STEM	1	3.3 %
	STEM	29	96.7 %
Study Subject	CS	12	40.0 %
	Non-CS	18	60.0 %
	Never	1	3.3 %
	Once	3	10.0 %
Experience [2]	Sometimes	12	40.0 %
	Repeatedly	4	13.3 %
	Regularly	10	33.3 %

Table 6.2 shows the analysis of results from Q1 to Q7. Most participants are males aged 21 to 30 and enrolled in Master of Science programs. Nearly all participants are from STEM fields, with the majority having a non-CS background. Additionally, nearly all participants have had more than one training experience in CS, CT, programming, or algorithmic problem-solving. Analysis of results from Q8 indicates that all participants reported low competence in at least one programming

language. Python was the most commonly cited language in which participants had above-average competence.

6.4.1. DIFFERENCES IN PERFORMANCE ON ALGORITHMIC ABSTRACTION TASKS

For RQ 1 (What are the differences in performance on algorithmic abstraction tasks for students with CS or non-CS backgrounds?), we analyzed the scores of tasks, with an independent Student's t-test being an additional measurement.

ANALYSIS ON THE SCORES

To examine whether the CS background plays an important role in the level of algorithmic abstraction skills, we started with a descriptive analysis of scores. Table 6.3 shows that participants with a CS background obtained a higher average score for every level of algorithmic abstraction skills. Meanwhile, the results show a trend in decrease in the average score from the *Problem Level* to the *Execution Level*, and participants obtain more than half of the scores on the *Problem Level*.

Table 6.3.: Descriptive statistics and Student's T-Test of scores for different algorithmic abstraction levels and total scores

Levels	**Domain**	**N**	**Mean**	**SD**[3]	**p-value**[4]
Problem Level	CS	12	0.840	0.1061	**0.009**
	Non-CS	18	0.714	0.1511	
Algorithm Level	CS	12	0.449	0.1598	0.139
	Non-CS	18	0.389	0.1349	
Program Level	CS	12	0.389	0.1436	0.338
	Non-CS	18	0.366	0.1489	
Execution Level	CS	12	0.286	0.1518	0.305
	Non-CS	18	0.257	0.1544	
All Levels	CS	12	0.471	0.0902	**0.041**
	Non-CS	18	0.412	0.0861	

Furthermore, we conducted an independent samples T-test to examine if the scores of the CS participants were significantly higher than those of the non-CS participants. Table 6.3 shows that there is a significant difference between the CS and the non-CS groups regarding the scores for the *Problem Level* and the sum of all levels, but not a significant difference regarding the *Algorithm Level*, *Program Level*, and the *Execution Level*.

THEMATIC ANALYSIS

At the *Problem Level*, participants demonstrated various understandings of the concept of an algorithm. Definitions of sorting varied, with participants applying it to different scenarios. Four non-CS participants defined sorting as a technique for grouping items. The remaining participants described sorting as a method for ordering items or as a technique that can be used for both grouping and ordering. Here are examples of each category from their answers. **Sorting as a technique to group things:** *"Sorting is dividing a larger group of items into different smaller groups, where they have a distinct feature.", "I work in administration for my football association, and I work a lot with excel where for each member there are some yes-no questions, for example, if they have completed some steps in the registration process. Then it is very easy for me to sort the whole list, by selecting the members that have not yet sent me a certain document, and I will contact them."* **Sorting as a technique to put things into a certain order:** *"Sorting is ordering items based on properties that can be compared to each other and said to be higher, lower, or equal.", "When creating backups of my files, going from smaller to bigger files ensures the biggest amount of files is uploaded, in case the internet cuts off halfway through."* **Sorting as a technique for both grouping things and putting things into a certain order:** *"Sorting is the act of arranging a set of elements in order or in groups based on some characteristic.", "I sort my clothes based on when I last wore them, so that I go through my whole closet, and do not repeat the same clothes too soon."*

At the *Algorithm Level*, we examined participants' approaches to algorithm design and the concreteness of their step-by-step plans. Seven CS-background participants considered the complexity of the problem, while the rest focused on problem-solving strategies or less relevant aspects for computational efficiency. Regarding the step-by-step plans, 11 participants (including four from a CS background) provided generic, strategic plans lacking specific operational steps. Most participants, however, offered concrete plans detailing basic algorithm operations such as *compare* and *swap*. **Consideration of the complexity of the problem:** *"The way to sort a deck of cards can be used to organize over 5,000 books, but I think it might be computationally intensive. especially as letters are involved and strings need to be compared ..."* **Consideration of the strategy for solving the problem or other aspects:** *"The way to sort a deck of cards cannot be used to organize over 5,000 books, it depends on which properties you divide them."* **Generic plan on a strategic level that does not contain specific operational steps:** *"In the desired order the cards are arranged numerically in increasing order. So I would just look for the card with a small number and keep it to the left."* **Concrete plan that includes the basic operation of the algorithm:** *"Start on the left side. Pick the first (leftmost) card. Compare it to the card to the right of it. If the card is higher than its rightside neighbour, swap them. Do this until you don't swap anything anymore and move on to the next card."*

At the *Program Level*, we analyzed the reflections of seven participants who did not implement the algorithm in Python. Their reflections were categorized into three main areas: time constraints, Python proficiency, and reliance on existing functions. Here are examples for each category. **Time:** *"It takes time for me to come up with a logic step by step and implement it."* **Python Proficiency:** *"I am not able to generate Python code spontaneously."* **Indication of using existing functions:** *"I would normally not sort anything myself since sorting algorithms are already implemented and available in most languages."*

At the *Execution Level*, eight participants, including five with a non-CS background, provided concrete plans for debugging and execution. The remaining participants gave more generic descriptions of their overall debugging and execution strategies. **Detailed plan for debugging and execution:** *“Write out the effects of the code in an example calculation, use a supporting code editor error analyzer", “Check syntax errors, debugging print statements, stepwise execution, logically thinking about the algorithm the code is following."* **Generic descriptions of overall strategy:** *“Run with test example" or "Write some tests with simple examples."*

6.4.2. Factors associated with one’s performance on algorithmic abstraction tasks

For RQ2, (which factors, among demographic factors, training experiences on CS, programming or algorithmic problem-solving, self-perception of proficiency with programming languages, and self-perception of mastery and necessity of algorithmic abstraction skills, are associated with one’s performance on algorithmic abstraction tasks?), we started with a basic statistical analysis of the answers on perceptions of mastery and the necessity of algorithmic abstraction skills. Table 6.5 shows that the variance is larger in the rating of the mastered level, with the CS participants achieving a higher mean and median. This indicates that the perceived mastered level of skills for CS participants is on average higher than that of the non-CS participants. Moreover, the results show that the lowest needed level of algorithmic abstraction skills is at the *Algorithm Level* for both CS and non-CS participants, and most participants deemed it needed to master the *Program Level* or even the *Execution Level*. Meanwhile, for the perceived needed level of algorithmic abstraction skills, both the mean and median of the level of algorithmic abstraction skills for CS participants is close to that of non-CS participants. However, the gap between the mastered level and the needed level is higher for non-CS participants than for CS participants. Moreover, both CS and non-CS participants indicated that the lowest level of algorithmic abstraction skills needed is the *Algorithm Level*.

Regarding how factors correlate with student performance, the results show no significant correlation between the total score across all tasks and demographic factors. Table 6.4 illustrates that these total scores correlate significantly with participants’ training experience and programming language proficiency. Meanwhile, there is no significant correlation between total scores and the perception of the mastered level and needed level of algorithmic abstraction skills, respectively. Furthermore, the analysis shows that training experience correlates significantly with programming experience, and study subjects correlate significantly with both training and programming experience.

6.5. Discussion and Conclusion

6.5.1. Limitations of this study

The study has several limitations. Firstly, the findings are based on a small sample size from a specific context in the Netherlands, limiting their generalizability. To enhance generalizability, replication in diverse contexts is crucial. Secondly, the sampling methods used are constrained by practical limitations, which also affect the study's broader applicability. Additionally, maintaining rigor in the thematic analysis of qualitative data is challenging, particularly due to the researcher's presence during data collection, which can influence participant responses and the interpretation of results.

6.5.2. Implications on algorithmic abstraction skills education

Based on the findings from RQ2, both CS and non-CS participants indicated the lowest perceived need for algorithmic abstraction skills at the *Algorithm Level*. Moreover, most participants expressed a necessity to master skills at the *Program Level* or the *Execution Level*. Given that nearly all participants are from STEM fields, we recognized the need to cultivate STEM students with at least algorithmic abstraction skills at the *Algorithm Level*, and potentially up to *Execution Level* depending on their backgrounds. Furthermore, non-CS participants show a larger gap between their perceived mastery and their perceived need compared to CS participants. Addressing these discrepancies is crucial for designing educational strategies that meet students' needs, requiring a careful examination of challenges and potential solutions.

The performance differences in algorithmic abstraction tasks between CS and non-CS participants highlight significant educational challenges. CS participants consistently scored higher across all levels compared to non-CS participants, with statistically significant differences observed mainly at the problem level and in the total scores across all levels. This suggests that CS participants may encounter fewer obstacles in understanding and reformulating algorithmic problems. This finding correlates with the observation that non-CS participants tend to define algorithm concepts differently, often diverging from computational contexts. In contrast to Aharoni's findings [129], where understanding was centered at the *Algorithm Level* of data structures, participants in this study typically grasp algorithmic abstraction at the Problem Level. Potential contributing factors include differences in the application context of the PGK framework, variations in the assessment instruments derived from the framework, and differences in sample composition compared to Aharoni's study [129].

There were also notable performance differences observed at other levels of algorithmic abstraction skills. At the *Algorithm Level*, CS participants demonstrate greater awareness of computational complexity in algorithm design compared to non-CS participants. This finding is consistent with a study [96] that explored the integration of computational thinking in engineering curricula, highlighting that non-CS students have less emphasis on mastering computational design compared to CS students. At the *Program Level*, concerns primarily revolve around time constraints and proficiency in programming languages when implementing algorithms. Moreover, only a minority of participants offered detailed plans for debugging and executing algorithms.

The findings from RQ2 suggest potential strategies for addressing challenges identified in the study. Both training experience and programming language proficiency significantly correlate with participants' performance, irrespective of their background. Building on these findings and considering the performance disparities revealed in RQ1, along with factors influencing performance identified in RQ2, it is recommended to prioritize training in CS, CT, programming, and algorithmic abstraction skills to enhance students' abilities in algorithmic abstraction. Furthermore, tailored training programs should address specific needs across different backgrounds. For STEM students in general, achieving proficiency at the *Algorithm Level* appears crucial. This can be facilitated through discussions on algorithmic concepts and introducing computational complexity, which is particularly beneficial for non-CS students. Educational initiatives aiming to enhance skills at the program level should account for learners' familiarity with programming languages and the time required for implementation. The challenges and impact of detailed debugging and execution plans across different domains warrant further investigation, particularly at the *Execution Level*.

6.5.3. Implications for future research

This research operationalized the PGK framework to investigate differences in student performance on algorithmic abstraction tasks, how demographic factors, training experience, programming proficiency, and students' perceived and actual mastery of these skills are correlated to their performance. Future work can build on this by focusing on several key areas: developing domain-specific curriculum modules tailored to the needs of various STEM fields to address gaps in understanding algorithm concepts and computational complexity; designing comprehensive training programs in computational thinking and programming for both CS and non-CS students to bridge the gap between needed and actual skill levels; conducting longitudinal studies to monitor the development of algorithmic abstraction skills and the effects of different educational interventions over time; fostering cross-disciplinary collaboration to integrate algorithmic thinking into non-CS curricula through interdisciplinary projects; and researching effective methods for teaching computational complexity to non-CS students, evaluating the benefits of introducing these concepts early in the curriculum.

Table 6.4.: Correlational analysis between the total score and demographic variables

		Gender	**Age**	**Education Level**	**Year of Study**	**Study Domain**	**Study Subject**	**Training Experience**	**Programming Language Proficiency**	**Perception on Mastered Level**	**Perception on Needed Level**
Score_All Levels	Person's r	0.143	-0.114	0.132	-0.286	-0.227	-0.316	0.402*	0.423*	0.223	-0.112
	p-value	0.450	0.548	0.487	0.126	0.227	0.089	0.028	0.020	0.237	0.556

Note: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 6.5.: Participants' perceptions of the mastered and level of algorithmic abstraction skills

	Mastered Level		**Needed Level**	
	CS	Non-CS	CS	Non-CS
Mean	3.58	2.44	3.50	3.17
Median	4	2	4	4
Standard Deviation	0.793	1.04	0.905	0.985
Minimum	2	1	2	2
Maximum	4	4	4	4

7

SUMMARY AND CONCLUSION

7.1. SUMMARY

With the prevalence of digital device usage in every aspect of our lives, Computational Thinking (CT) is an increasingly important skill set to obtain. Despite the importance of CT skills, there is limited attention paid to them when it comes to the integration and assessment of CT in higher education. In this thesis we approached several questions considering the existing level of knowledge on the topic of CT assessment in higher education, the current practices in CT assessment in higher education, teachers' understanding of CT and their perceptions of integrating CT into practices. Furthermore, we analysed the integration of CT into teaching practices and the factors influencing teachers' intention to embed it into their teaching practices. This dissertation explores the integration of CT into Higher Education curricula and CT assessment in higher education.

In this dissertation, **Part I** has been designed to investigate the state-of-the-art of CT assessment in higher education in the literature, with which we gained a theoretical understanding regarding the topic of CT assessment in higher education. Aspects of assessment design, including a) methods, b) constructs, and c) contexts, and empirical knowledge on CT assessment in higher education regarding those aspects, were investigated through a systematic umbrella review and a systematic review in Chapter 2 and Chapter 3, respectively.

To facilitate designing effective assessment, we examine how in practice CT has been integrated and what needs to be assessed. Based on findings from **Part I**, Chapter 4 and Chapter 5 in **Part II** (addressing **Objective II**), explored the CT integration in higher education, especially the integration of CT into Master's level engineering curricula at Delft University of Technology, and analysed teacher's perceptions of CT in Engineering Education and their intentions to integrate it into teaching.

Taking together the research investigated aspects of CT assessment and practical information from CT integration in higher education from **Part I** and **Part II**, and **Part III** aiming at (**Objective 3**) the focused design of assessment materials to measure algorithmic abstraction skills (one of the core components of CT skills).

7.2. Main findings

7.2.1. The state-of-the-art of CT assessment in higher education (Part I)

The systematic umbrella review in Chapter 2 provides insights from 11 reviews on computational thinking (CT) assessment in higher education. The findings highlight various dimensions related to assessment methods, constructs, and contexts. Most reviews do not solely focus on CT assessment in higher education. While a combination of different assessment methods is recommended, clear guidelines for effective assessment design are still lacking. We identified 120 unique constructs and about 10 types of assessment methods. In addition to that, a combined usage of different assessment methods for measuring CT skills is recommended from the findings.

Additionally, the systematic review in Chapter 3 analyzed 47 empirical studies on CT assessment in higher education, published between 1986 and 2021. Most assessments were summative, and conducted in STEM fields at the undergraduate level. These studies assessed CT constructs using various naming and categorization schemes, employing both existing and newly designed assessment tools. Moreover, most of the studies apply manual teacher assessment in the form of traditional test & exams.

7.2.2. CT integration in higher education (Part II)

7

In Chapter 4, we examined the integration of computational thinking (CT) into the Master's level Engineering curriculum at Delft University of Technology. Our findings reveal that CT is well-integrated into the curriculum, with lectures serving as the primary educational method and programming assignments being the most common assessment tool. Additionally, we observed that understanding the context and patterns in problems and solutions is crucial across different courses and engineering disciplines.

In Chapter 5, we surveyed 38 teachers in tertiary education in the Netherlands about their (1) perceptions of computational thinking (CT) and (2) intentions to integrate CT into their teaching. We found that teachers generally lack a clear understanding of the relationship between CT and Computer Science, have limited training in CT, and hold varying opinions on when and which CT constructs should be integrated into different subjects. Additionally, there is a strong positive correlation between teachers' performance expectancy, attitude toward CT, and their intention to incorporate CT into their teaching. To promote CT integration in higher education, our findings highlight the need for enhanced teacher training programs focused on CT and its connection to Computer Science.

7.2.3. CT assessment implementation in higher education (Part III)

In Chapter 6, we designed an assessment tool for a learning task about sorting algorithms and conducted a study to evaluate students' abstraction skills in understanding sorting algorithms. Our findings indicate that both CS and non-CS students need abstraction skills at least at the *Algorithm Level*, with many requiring skills at the *Program Level* or the *Execution Level*. This is particularly crucial for STEM students, emphasizing the need for tailored educational approaches for non-CS students, who exhibit a larger gap between their perceived and required mastery levels. Training and programming experience correlate strongly with performance, underscoring the importance of targeted training in computational thinking, programming, and algorithmic abstraction. Such training should be discipline-specific, with a minimum requirement at the *Algorithm Level*.

7.3. General Discussion

7.3.1. CT assessment in higher education needs more empirical research

As suggested by the results and findings in **Part I** through Chapters 2 and 3, the topic of CT assessment in higher education encompasses a wide range of sub-topics. CT has been defined, operationalized, and implemented differently across various contexts, as highlighted in **Part II** through Chapters 4 and 5, which examine curricula and teachers' perceptions and understandings from diverse domains. These variations introduce complexities when designing effective assessment instruments for measuring CT skills.

Designing effective assessments requires carefully considering what, how, when, whom, where, and why to assess. Empirical experiences provide valuable insights into what has worked and potential directions for future exploration. While the literature lists various constructs to assess and available assessment methods in Chapters 2 and 3, there are limited empirical studies investigating CT assessment in higher education. Consequently, the generalizability and applicability of existing assessment instruments and practices remain uncertain. More empirical research is needed to refine these instruments and ensure they are effective across different educational contexts.

7.3.2. Not everyone needs the same set of CT skills at the same level

As demonstrated in Chapters 4 and 5, while computational thinking (CT) is essential for everyone, its integration requires careful consideration of both the content and method. Not all individuals need to master every CT construct, and this variability should be taken into account when designing assessments. It is crucial to identify the specific constructs that need to be assessed for different backgrounds before applying general assessment constructs.

Chapter 2 reveals that there are 120 unique constructs measured in CT assessments, with abstraction and algorithmic thinking being among the most frequently mentioned. Our findings in Chapters 4 and 5 further emphasize that algorithms and abstraction are critical constructs to master, particularly in STEM domains. Based on these insights, we designed the study in Chapter 6, which reinforced the understanding that not everyone requires the same set of CT skills at the same level. This realization underscores the importance of thoughtfully integrating CT into curricula and designing appropriate CT assessment instruments tailored to different needs and contexts.

7.3.3. Training is needed for both Teachers and Students for promoting CT for all

The findings from this dissertation underscore the critical need for targeted training programs to effectively promote CT across higher education. As CT becomes increasingly essential in our digital age, its integration into curricula requires comprehensive training for both educators and students.

Our research reveals significant gaps in teachers' understanding of CT at a basic level, particularly in distinguishing it from traditional Computer Science concepts. The survey of tertiary educators in Chapter 5 highlighted that many educators lack a clear grasp of CT's relevance across disciplines, compounded by limited formal training in CT. To address this, tailored training programs should be developed to deepen teachers' understanding of CT and provide strategies for incorporating it into their curricula. Such professional development would empower teachers and ensure consistent and effective CT education for students across various fields.

The need for student training is equally vital, particularly as CT becomes a foundational skill across disciplines. In Chapter 6, the study evaluating students' abstraction skills in sorting algorithms revealed significant proficiency gaps, especially among non-CS students. These students often struggle with the abstraction levels necessary for understanding and applying CT concepts, indicating a need for focused, discipline-specific training programs. Targeted educational interventions should be designed to build foundational CT skills, for example in algorithmic abstraction, tailored to the context of their specific fields. By addressing the training needs of both teachers and students, we can foster a more comprehensive and inclusive approach to CT education in higher education.

8

SAMENVATTING

Met de prevalentie van het gebruik van digitale apparaten in elk aspect van ons leven, is Computational Thinking (CT) een steeds belangrijkere vaardigheid om te verwerven. Ondanks het belang van CT-vaardigheden, wordt er weinig aandacht aan besteed als het gaat om de integratie en beoordeling van CT in het hoger onderwijs. In dit proefschrift benaderden we verschillende vragen met betrekking tot het bestaande kennisniveau over het onderwerp CT-beoordeling in het hoger onderwijs, de huidige praktijken in CT-beoordeling in het hoger onderwijs, het begrip van docenten van CT en hun percepties van het integreren van CT in praktijken. Verder analyseerden we de integratie van CT in onderwijspraktijken en de factoren die van invloed zijn op de intentie van docenten om het in hun onderwijspraktijken op te nemen. Dit proefschrift onderzoekt de integratie van CT in curricula van het hoger onderwijs en CT-beoordeling in het hoger onderwijs.

In dit proefschrift is **Deel I** ontworpen om de stand van zaken van CT-beoordeling in het hoger onderwijs in de literatuur te onderzoeken, waarmee we een theoretisch begrip hebben gekregen met betrekking tot het onderwerp CT-beoordeling in het hoger onderwijs. Aspecten van beoordelingsontwerp, waaronder a) methoden, b) constructies en c) contexten, en empirische kennis over CT-beoordeling in het hoger onderwijs met betrekking tot die aspecten, werden onderzocht via een systematische paraplubeoordeling en een systematische beoordeling in respectievelijk Hoofdstuk 2 en Hoofdstuk 3.

Om het ontwerpen van effectieve beoordeling te vergemakkelijken, onderzoeken we hoe CT in de praktijk is geïntegreerd en wat er moet worden beoordeeld. Op basis van bevindingen uit **Deel I**, Hoofdstuk 4 en Hoofdstuk 5 in **Deel II** (met betrekking tot **Doelstelling II**), onderzochten we de CT-integratie in het hoger onderwijs, met name de integratie van CT in masteropleidingen voor ingenieurs aan de Technische Universiteit Delft, en analyseerden we de percepties van docenten over CT in het ingenieursonderwijs en hun intenties om het te integreren in het lesgeven.

Door het onderzoek samen te voegen werden aspecten van CT-beoordeling en praktische informatie uit CT-integratie in het hoger onderwijs uit **Deel I** en **Deel II**, en **Deel III** onderzocht, met als doel (**Doelstelling 3**) het gericht ontwerpen van beoordelingsmaterialen om algoritmische abstractievaardigheden te meten (een van de kerncomponenten van CT-vaardigheden).

8.1. ALGEMENE DISCUSSIE

8.1.1. CT-BEOORDELING IN HET HOGER ONDERWIJS HEEFT MEER EMPIRISCH ONDERZOEK NODIG

Zoals gesuggereerd door de resultaten en bevindingen in **Deel I** tot en met Hoofdstukken 2 en 3, omvat het onderwerp CT-beoordeling in het hoger onderwijs een breed scala aan subonderwerpen. CT is in verschillende contexten op verschillende manieren gedefinieerd, geoperationaliseerd en geïmplementeerd, zoals benadrukt in **Deel II** tot en met Hoofdstukken 4 en 5, die curricula en de percepties en inzichten van docenten uit diverse domeinen onderzoeken. Deze variaties introduceren complexiteiten bij het ontwerpen van effectieve beoordelingsinstrumenten voor het meten van CT-vaardigheden.

Het ontwerpen van effectieve beoordelingen vereist zorgvuldig overwegen wat, hoe, wanneer, wie, waar en waarom beoordeeld moet worden. Empirische ervaringen bieden waardevolle inzichten in wat heeft gewerkt en mogelijke richtingen voor toekomstige verkenning. Hoewel de literatuur verschillende constructen voor beoordeling en beschikbare beoordelingsmethoden in hoofdstukken 2 en 3 vermeldt, zijn er beperkte empirische studies die CT-beoordeling in het hoger onderwijs onderzoeken. Bijgevolg blijven de generaliseerbaarheid en toepasbaarheid van bestaande beoordelingsinstrumenten en -praktijken onzeker. Er is meer empirisch onderzoek nodig om deze instrumenten te verfijnen en ervoor te zorgen dat ze effectief zijn in verschillende onderwijscontexten.

8.1.2. NIET IEDEREEN HEEFT DEZELFDE SET CT-VAARDIGHEDEN OP HETZELFDE NIVEAU NODIG

Zoals aangetoond in hoofdstukken 4 en 5, is computationeel denken (CT) essentieel voor iedereen, maar vereist de integratie ervan zorgvuldige overweging van zowel de inhoud als de methode. Niet alle individuen hoeven elk CT-construct onder de knie te krijgen en deze variabiliteit moet in acht worden genomen bij het ontwerpen van beoordelingen. Het is cruciaal om de specifieke constructen te identificeren die moeten worden beoordeeld voor verschillende achtergronden voordat algemene beoordelingsconstructen worden toegepast.

Hoofdstuk 2 onthult dat er 120 unieke constructen worden gemeten in CT-beoordelingen, waarbij abstractie en algoritmisch denken tot de meest genoemde behoren. Onze bevindingen in hoofdstukken 4 en 5 benadrukken verder dat algoritmen en abstractie cruciale constructen zijn om onder de knie te krijgen, met name in STEM-domeinen. Op basis van deze inzichten hebben we de studie in hoofdstuk 6 ontworpen, die het begrip versterkte dat niet iedereen dezelfde set CT-vaardigheden op hetzelfde niveau nodig heeft. Deze realisatie onderstreept het belang van het zorgvuldig integreren van CT in curricula en het ontwerpen van geschikte CT-beoordelingsinstrumenten die zijn afgestemd op verschillende behoeften en contexten.

8.1.3. TRAINING IS NODIG VOOR ZOWEL DOCENTEN ALS STUDENTEN OM CT VOOR IEDEREEN TE PROMOTEN

De bevindingen van dit proefschrift benadrukken de cruciale behoefte aan gerichte trainingsprogramma's om CT effectief te promoten in het hoger onderwijs. Omdat CT steeds belangrijker wordt in ons digitale tijdperk, vereist de integratie ervan in curricula uitgebreide training voor zowel docenten als studenten.

Uit ons onderzoek blijkt dat er aanzienlijke hiaten zijn in het begrip van CT door docenten op basisniveau, met name in het onderscheiden van traditionele computerwetenschapsconcepten. Het onderzoek onder docenten in het hoger onderwijs in hoofdstuk 5 benadrukte dat veel docenten geen duidelijk begrip hebben van de relevantie van CT in verschillende disciplines, wat nog eens wordt verergerd door de beperkte formele training in CT. Om dit aan te pakken, moeten er op maat gemaakte trainingsprogramma's worden ontwikkeld om het begrip van CT door docenten te verdiepen en strategieën te bieden om het in hun curricula op te nemen. Dergelijke professionele ontwikkeling zou docenten meer macht geven en consistente en effectieve CT-educatie voor studenten in verschillende vakgebieden garanderen.

De behoefte aan studententraining is net zo belangrijk, met name omdat CT een fundamentele vaardigheid wordt in verschillende disciplines. In hoofdstuk 6 onthulde de studie die de abstractievaardigheden van studenten in sorteeralgoritmen evalueerde, aanzienlijke vaardigheidshiaten, met name onder niet-CS-studenten. Deze studenten worstelen vaak met de abstractieniveaus die nodig zijn om CT-concepten te begrijpen en toe te passen, wat wijst op de behoefte aan gerichte, disciplinespecifieke trainingsprogramma's. Gerichte educatieve interventies moeten worden ontworpen om fundamentele CT-vaardigheden op te bouwen, bijvoorbeeld in algoritmische abstractie, afgestemd op de context van hun specifieke vakgebieden. Door in te spelen op de trainingsbehoeften van zowel docenten als studenten, kunnen we een uitgebreidere en inclusievere benadering van CT-onderwijs in het hoger onderwijs bevorderen.

Bibliography

[1] J. M. Wing. "Computational thinking". In: *Communications of the ACM* 49.3 (2006), pp. 33–35.

[2] A. Kamp. *Engineering education in the rapidly changing world: Rethinking the vision for higher engineering education*. TU Delft, Faculty of Aerospace Engineering, 2016.

[3] *EU cooperation in education and training (ET 2020) | EUR-Lex — eur-lex.europa.eu*. https://eur-lex.europa.eu/EN/legal-content/summary/eu-cooperation-in-education-and-training-et-2020.html. [Accessed 17-08-2024].

[4] J. A. Lyon and A. J. Magana. "Computational thinking in higher education: A review of the literature". In: *Computer Applications in Engineering Education* 28.5 (Sept. 2020), pp. 1174–1189. ISSN: 10990542. DOI: 10.1002/cae.22295. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/cae.22295%20https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.22295%20https://onlinelibrary.wiley.com/doi/10.1002/cae.22295.

[5] A. Yadav, S. Gretter, J. Good, and T. Mclean. "Computational Thinking in Teacher Education". In: *Emerging Research, Practice, and Policy on Computational Thinking* (2017), pp. 205–220. DOI: 10.1007/978-3-319-52691-1_13. URL: https://link.springer.com/chapter/10.1007/978-3-319-52691-1_13.

[6] A. M. Thijs, P. H. G. Fisser, and M. van der Hoeven. *21e eeuwse vaardigheden in het curriculum van het funderend onderwijs [21st century skills in the curriculum of Dutch primary and lower secondary education]*. SLO, nationaal expertisecentrum leerplanontwikkeling, 2014.

[7] C. Lu, R. Macdonald, B. Odell, V. Kokhan, C. Demmans Epp, and M. Cutumisu. "A scoping review of computational thinking assessments in higher education". In: *Journal of Computing in Higher Education* 0123456789 (Jan. 2022). ISSN: 1042-1726. DOI: 10.1007/s12528-021-09305-y. URL: https://doi.org/10.1007/s12528-021-09305-y.

[8] J. A. Lyon and A. J. Magana. "Computational thinking in higher education: A review of the literature". In: *Computer Applications in Engineering Education* 28.5 (2020), pp. 1174–1189.

[9] A. Yadav, J. Good, J. Voogt, and P. Fisser. "Computational thinking as an emerging competence domain". In: *Competence-based vocational and professional education: Bridging the worlds of work and education* (2017), pp. 1051–1067.

[10] M. van Onderwijs and C. en Wetenschap. *Types of reports of the Dutch Inspectorate of Education-Inspection-Inspectorate of Education*. Ministerie van Onderwijs, Cultuur en Wetenschap, 2016.

[11] Y.-P. Hsiao, G. van de Watering, M. Heitbrink, H. Vlas, and M.-S. Chiu. “Ensuring bachelor’s thesis assessment quality: a case study at one Dutch research university”. In: *Higher Education Evaluation and Development* 18.1 (2024), pp. 2–16.

[12] X. Tang, Y. Yin, Q. Lin, R. Hadad, and X. Zhai. “Assessing computational thinking: A systematic review of empirical studies”. In: *Computers and Education* 148 (Apr. 2020), p. 103798. ISSN: 03601315. DOI: 10.1016/j.compedu.2019.103798. URL: https://doi.org/10.1016/j.compedu.2019.103798.

[13] I. De Jong and J. Jeuring. “Computational Thinking Interventions in Higher Education: A Scoping Literature Review of Interventions Used to Teach Computational Thinking”. In: *ACM International Conference Proceeding Series*. Vol. 10. 20. New York, NY, USA: ACM, 2020. ISBN: 9781450389211. DOI: 10.1145/3428029.3428055. URL: https://doi.org/10.1145/3428029.3428055.

[14] A. L. S. O. De Araujo, W. L. Andrade, and D. D. Serey Guerrero. “A systematic mapping study on assessing computational thinking abilities”. In: *Proceedings - Frontiers in Education Conference, FIE* 2016-Novem (2016). ISSN: 15394565. DOI: 10.1109/FIE.2016.7757678.

[15] J. Perrenet and E. Kaasenbrood. “Levels of abstraction in students’ understanding of the concept of algorithm: the qualitative perspective”. In: *ACM SIGCSE Bulletin* 38.3 (2006), pp. 270–274.

[16] J. Perrenet, J. F. Groote, and E. Kaasenbrood. “Exploring students’ understanding of the concept of algorithm: levels of abstraction”. In: *ACM SIGCSE Bulletin* 37.3 (2005), pp. 64–68.

[17] X. Zhang, F. Aivaloglou, and M. Specht. “A Systematic Umbrella Review on Computational Thinking Assessment in Higher Education.” In: *European Journal of STEM Education* 9.1 (2024), p. 2.

[18] J. Wing. “Research notebook: Computational thinking—What and why”. In: *The link magazine* 6 (2011), pp. 20–23.

[19] P. J. Denning and M. Tedre. *Computational thinking*. The MIT press essential knowledge series. Cambridge, Massachusetts: The MIT Press, 2019. ISBN: 9780262536561.

[20] M. Cutumisu, C. Adams, and C. Lu. “A Scoping Review of Empirical Research on Recent Computational Thinking Assessments”. In: *Journal of Science Education and Technology* 28.6 (Dec. 2019), pp. 651–676. ISSN: 15731839. DOI: 10.1007/s10956-019-09799-3.

[21] X. Zhang and M. Specht. “A Review of Reviews on Computational Thinking Assessment in Higher Education”. en. In: *CTE-STEM 2022 conference* (June 2022), 2022: APSCE CTESTEM conference. DOI: 10.34641/CTESTEM.2022.472. URL: https://proceedings.open.tudelft.nl/cte-stem2022/article/view/472 (visited on 09/07/2022).

[22] K. Brennan and M. Resnick. *New Frameworks for Studying and Assessing the Development of Computational Thinking*. Vol. Proceedings of the 2012 Annual Meeting of the American Educational Research Association. Vancouver, Apr. 2012. URL: http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf.

[23] D. Weintrop, E. Beheshti, M. Horn, K. Orton, K. Jona, L. Trouille, and U. Wilensky. “Defining Computational Thinking for Mathematics and Science Classrooms”. en. In: *Journal of Science Education and Technology* 25.1 (Feb. 2016), pp. 127–147. ISSN: 1059-0145, 1573-1839. DOI: 10.1007/s10956-015-9581-5. URL: http://link.springer.com/10.1007/s10956-015-9581-5 (visited on 09/07/2022).

[24] S. Grover and R. D. Pea. “Computational Thinking: A Competency Whose Time Has Come”. In: *Computer Science Education*. Dec. 2017. URL: 10.5040/9781350057142.ch-003.

[25] T.-C. Hsu, S.-C. Chang, and Y.-T. Hung. “How to learn and how to teach computational thinking: Suggestions based on a review of the literature”. en. In: *Computers & Education* 126 (Nov. 2018), pp. 296–310. ISSN: 03601315. DOI: 10.1016/j.compedu.2018.07.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0360131518301799 (visited on 09/07/2022).

[26] C. Angeli and M. Giannakos. “Computational thinking education: Issues and challenges”. en. In: *Computers in Human Behavior* 105 (Apr. 2020), p. 106185. ISSN: 07475632. DOI: 10.1016/j.chb.2019.106185. URL: https://linkinghub.elsevier.com/retrieve/pii/S0747563219303978 (visited on 09/07/2022).

[27] X. Li and C. Gu. “Teaching reform of programming basic course based on SPOC blended teaching method”. In: *2020 15th International Conference on Computer Science & Education (ICCSE)*. Delft, Netherlands: IEEE, Aug. 2020, pp. 411–415. ISBN: 9781728172675. DOI: 10.1109/ICCSE49874.2020.9201802. URL: https://ieeexplore.ieee.org/document/9201802/ (visited on 09/07/2022).

[28] J. M. Wing. “Computational thinking”. en. In: *Communications of the ACM* 49.3 (Mar. 2006), pp. 33–35. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/1118178.1118215. URL: https://dl.acm.org/doi/10.1145/1118178.1118215 (visited on 09/07/2022).

[29] C. Selby and J. Woollard. *Computational thinking: the developing definition*. Tech. rep. University of Southampton, 2013.

[30] N. O. Ezeamuzie and J. S. C. Leung. "Computational Thinking Through an Empirical Lens: A Systematic Review of Literature". In: *Journal of Educational Computing Research* (2021), p. 073563312110331. ISSN: 0735-6331. DOI: 10.1177/07356331211033158.

[31] H. I. Hasesk, U. Ilic, H. İ. Haseskİ, and U. İlİc. "An Investigation of the Data Collection Instruments Developed to Measure Computational Thinking". In: *Informatics in Education* 18.2 (2019), pp. 297–319. ISSN: 16485831. DOI: 10.15388/infedu.2019.14.

[32] V. V. Vinu Varghese and V. G. Renumol. "Assessment methods and interventions to develop computational thinking - A literature review". In: *2021 International Conference on Innovative Trends in Information Technology, ICITIIT 2021* (2021). DOI: 10.1109/ICITIIT51526.2021.9399606.

[33] E. Poulakis and P. Politis. "Computational Thinking Assessment: Literature Review". In: *Research on E-Learning and ICT in Education* (2021), pp. 111–128. DOI: 10.1007/978-3-030-64363-8_7.

[34] E. Taslibeyaz, E. Kursen, and S. Karaman. "How to Develop Computational Thinking: A Systematic Review of Empirical Studies". In: *Informatics in Education* 19.4 (2020), pp. 701–719. ISSN: 16485831. DOI: 10.15388/INFEDU.2020.30.

[35] P. Fusar-Poli and J. Radua. "Ten simple rules for conducting umbrella reviews". In: *BMJ Ment Health* 21.3 (2018), pp. 95–100.

[36] J. Taylor. "Doing Your Literature Review-Traditional and Systematic Techniques Jill K Jesson Doing Your Literature Review-Traditional and Systematic Techniques, Lydia Matheson Fiona M Lacey£ 20.99 192pp 9781848601543 1848601549 [Formula: see text]." In: *Nurse Researcher* 19.4 (2012), pp. 45–45.

[37] J. Zhu and W. Liu. "A tale of two databases: the use of Web of Science and Scopus in academic papers". In: *Scientometrics* 123.1 (Feb. 2020), pp. 321–335. ISSN: 15882861. DOI: 10.1007/s11192-020-03387-8. arXiv: 2002.02608. URL: https://link.springer.com/article/10.1007/s11192-020-03387-8.

[38] J. Biggs. "Enhancing Teaching Through Constructive Alignment". In: *Higher Education*. Vol. 32. 1996, 347 ~ 364.

[39] F. R. Sullivan and J. Heffernan. "Robotic Construction Kits as Computational Manipulatives for Learning in the STEM Disciplines". In: *Journal of Research on Technology in Education* 48.2 (2016), 105~128.

[40] J. H. McMillan. *Classroom assessment : principles and practice for effective standards-based instruction*. Pearson, 2013.

[41] C. Lu, R. Macdonald, B. Odell, V. Kokhan, C. Demmans Epp, and M. Cutumisu. "A scoping review of computational thinking assessments in higher education". In: *Journal of Computing in Higher Education* 34.2 (2022), pp. 416–461.

8

[42] K. Brennan and M. Resnick. "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. 2012, p. 25.

[43] R. J. Mislevy and G. D. Haertel. "Implications of evidence-centered design for educational testing". In: *Educational measurement: issues and practice* 25.4 (2006), pp. 6–20.

[44] D. Krathwohl. "A Revision Bloom's Taxonomy: An Overview". In: *Theory into Practice* (2002).

[45] I. Altanis and S. Retalis. "A multifaceted students' performance assessment framework for motion-based game-making projects with Scratch". In: *Educational Media International* 56.3 (July 2019), pp. 201–217. ISSN: 14695790. DOI: 10.1080/09523987.2019.1669876. URL: https://www.tandfonline.com/doi/abs/10.1080/09523987.2019.1669876.

[46] J. Krugel and P. Hubwieser. "Computational thinking as springboard for learning object-oriented programming in an interactive MOOC". In: *IEEE Global Engineering Education Conference, EDUCON*. IEEE Computer Society, June 2017, pp. 1709–1712. ISBN: 9781509054671. DOI: 10.1109/EDUCON.2017.7943079.

[47] D. Kafura, A. C. Bart, and B. Chowdhury. "Design and preliminary results from a computational thinking course". In: *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*. Vol. 2015-June. New York, NY, USA: ACM, 2015, pp. 63–68. ISBN: 9781450334402. DOI: 10.1145/2729094.2742593. URL: http://dx.doi.org/10.1145/2729094.2742593.

[48] A. R. Basawapatna and A. Repenning. "Cyberspace meets brick and mortar: An investigation into how students engage in peer to peer feedback using both cyberlearning and physical infrastructures". In: *ITiCSE'10 - Proceedings of the 2010 ACM SIGCSE Annual Conference on Innovation and Technology in Computer Science Education*. New York, New York, USA: ACM Press, 2010, pp. 184–188. ISBN: 9781605588209. DOI: 10.1145/1822090.1822143.

[49] D. Pung, D. P. T. Bui, and P. Janssen. "Automated Grading of Parametric Modelling Assignments". In: *RE: Anthropocene, Design in the Age of Humans - Proceedings of the 25th International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA 2020*. Vol. 2. 2020, pp. 323–332. ISBN: 9789887891741.

[50] M. Alrashidi, M. Gardner, and V. Callaghan. "Evaluating the use of pedagogical virtual machine with augmented reality to support learning embedded computing activity". In: *ACM International Conference Proceeding Series*. Vol. Part F1278. Association for Computing Machinery, Feb. 2017, pp. 44–50. ISBN: 9781450348096. DOI: 10.1145/3057039.3057088. URL: http://dx.doi.org/10.1145/3057039.3057088.

8

[51] A. P. Ambrósio, C. Xavier, and F. Georges. “Digital ink for cognitive assessment of computational thinking”. In: *Proceedings - Frontiers in Education Conference, FIE*. Vol. 2015-Febru. February. Institute of Electrical and Electronics Engineers Inc., Feb. 2015. DOI: 10.1109/FIE.2014.7044237.

[52] A. L. S. O. Araujo, J. S. Santos, W. L. Andrade, D. D. Guerrero, and V. Dagiene. “Exploring computational thinking assessment in introductory programming courses”. In: *Proceedings - Frontiers in Education Conference, FIE*. Vol. 2017-Octob. Institute of Electrical and Electronics Engineers Inc., Dec. 2017, pp. 1–9. ISBN: 9781509059195. DOI: 10.1109/FIE.2017.8190652.

[53] K. D. Boom, M. Bower, A. Arguel, J. Siemon, and A. Scholkmann. “Relationship between computational thinking and a measure of intelligence as a general problem-solving ability”. In: *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*. Vol. 18. New York, NY, USA: ACM, 2018, pp. 206–211. ISBN: 9781450357074. DOI: 10.1145/3197091.3197104. URL: https://doi.org/10.1145/3197091.3197104.

[54] C. Cabo and R. D. Lansiquot. “Integrating creative writing and computational thinking to develop interdisciplinary connections”. In: *ASEE Annual Conference and Exposition, Conference Proceedings*. Vol. 2016-June. American Society for Engineering Education, June 2016. DOI: 10.18260/p.25795.

[55] C. Cachero, P. Barra, S. Melia, and O. Lopez. “Impact of Programming Exposure on the Development of Computational Thinking Capabilities: An Empirical Study”. In: *IEEE Access* 8 (2020), pp. 72316–72325. ISSN: 21693536. DOI: 10.1109/ACCESS.2020.2987254.

[56] S. Y. Choi. “Development of an instructional model based on constructivism for fostering computational thinking”. In: *International Journal of Innovative Technology and Exploring Engineering* 8.3C (2019), pp. 381–385. ISSN: 22783075.

[57] L. Corral. “Experience report of a software development course in a faculty of fine arts”. In: *CEUR Workshop Proceedings*. Vol. 2190. 2018. URL: http://ceur-ws.org.

[58] J. G. P. Francisco and A. R. López. “Relationship of knowledge to learn in programming methodology and evaluation of computational thinking”. In: *TEEM ’16: Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality*. Vol. 02-04-Nove. New York, NY, USA: ACM, 2016, pp. 73–77. ISBN: 9781450347471. DOI: 10.1145/3012430.3012499. URL: http://dx.doi.org/10.1145/3012430.3012499.

[59] L. Gouws, K. Bradshaw, and P. Wentworth. “First year student performance in a test for computational thinking”. In: *ACM International Conference Proceeding Series*. New York, New York, USA: ACM Press, 2013, pp. 271–277. ISBN: 9781450321129. DOI: 10.1145/2513456.2513484. URL: http://dx.doi.org/10.1145/2513456.2513484.

[60] H. Y. Hou, S. Agrawal, and C. F. Lee. “Computational thinking training with technology for non-information undergraduates”. In: *Thinking Skills and Creativity* 38 (Dec. 2020), p. 100720. ISSN: 18711871. DOI: 10.1016/j.tsc.2020.100720.

[61] C. Kazimoglu, M. Kiernan, L. Bacon, and L. MacKinnon. “Learning programming at the computational thinking level via digital game-play”. In: *Procedia Computer Science*. Vol. 9. Elsevier, Jan. 2012, pp. 522–531. DOI: 10.1016/j.procs.2012.04.056.

[62] B. Kim, T. Kim, and J. Kim. “Paper-and-pencil programming strategy toward computational thinking for non-majors: Design your solution”. In: *Journal of Educational Computing Research* 49.4 (Apr. 2014), pp. 437–459. ISSN: 15414140. DOI: 10.2190/EC.49.4.b. URL: https://journals.sagepub.com/doi/10.2190/EC.49.4.b.

[63] K. H. Koh, H. Nickerson, A. Repenning, and A. Basawapatna. “Early validation of computational thinking pattern analysis”. In: *ITICSE 2014 - Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference*. 2014, pp. 213–218. ISBN: 9781450328333. DOI: 10.1145/2591708.2591724. URL: http://dx.doi.org/10.1145/2591708.2591724.

[64] Y. Lee and J. Cho. “Quantifying the effects of programming education on students’ knowledge representation and perception in computational thinking”. In: *International Journal of Innovation, Creativity and Change* 9.4 (2019), pp. 27–38. ISSN: 22011323. DOI: 10.31219/osf.io/zhetf. URL: https://osf.io/preprints/zhetf/.

[65] S. Leela, S. Chookaew, and P. Nilsook. “An effective microlearning approach using living book to promote vocational students’ computational thinking”. In: *ACM International Conference Proceeding Series*. New York, NY, USA: ACM, 2019, pp. 25–29. ISBN: 9781450372206. DOI: 10.1145/3369199.3369200. URL: https://doi.org/10.1145/3369199.3369200.

[66] T. Li and T. Wang. “A Unified Approach to Teach Computational Thinking for First Year Non–CS Majors in an Introductory Course”. In: *IERI Procedia* 2 (2012), pp. 498–503. ISSN: 22126678. DOI: 10.1016/j.ieri.2012.06.123.

[67] X. Li and C. Gu. “Teaching reform of programming basic course based on SPOC blended teaching method”. In: *15th International Conference on Computer Science and Education, ICCSE 2020*. Institute of Electrical and Electronics Engineers Inc., Aug. 2020, pp. 411–415. ISBN: 9781728172675. DOI: 10.1109/ICCSE49874.2020.9201802.

[68] Y. T. Lin, M. K.-C. Yeh, and H. L. Hsieh. “Teaching computer programming to science majors by modelling”. In: *Computer Applications in Engineering Education* 29.1 (Jan. 2021), pp. 130–144. ISSN: 10990542. DOI: 10.1002/cae.22247. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/cae.22247%20https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.22247%20https://onlinelibrary.wiley.com/doi/10.1002/cae.22247.

[69] S. Maria and T. Tsiatsos. "Case Study: Integrating computational thinking into the introductory course of computer science via the use of the programming language python". In: *Advances in Intelligent Systems and Computing*. Vol. 725. Springer Verlag, 2018, pp. 531–541. ISBN: 9783319751740. DOI: 10.1007/978-3-319-75175-7_52.

[70] N. V. Mendoza Diaz, R. Meier, D. A. Trytten, and S. Yoon Yoon. "Computational Thinking Growth during a First-Year Engineering Course". In: *Proceedings - Frontiers in Education Conference, FIE*. Vol. 2020-Octob. Institute of Electrical and Electronics Engineers Inc., Oct. 2020. ISBN: 9781728189611. DOI: 10.1109/FIE44824.2020.9274250.

[71] M. Meysenburg, T. D. Brooks, R. Burks, E. Doyle, and T. Frey. "DIVAS: Outreach to the natural sciences through image processing". In: *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. Vol. 2018-Janua. New York, NY, USA: ACM, 2018, pp. 777–782. ISBN: 9781450351034. DOI: 10.1145/3159450.3159537. URL: https://doi.org/10.1145/3159450.3159537.

[72] S. Mishra and S. Iyer. "An exploration of problem posing-based activities as an assessment tool and as an instructional strategy". In: *Research and Practice in Technology Enhanced Learning* 10.1 (June 2015), pp. 1–19. ISSN: 17937078. DOI: 10.1007/s41039-015-0006-0. URL: https://telrp.springeropen.com/articles/10.1007/s41039-015-0006-0.

[73] N. A. Nunez, G. Cornejo-Meza, and S. A. Sanchez. "Comparing computational thinking skills of engineering students in urban and rural areas of Peru". In: *2020 IEEE ANDESCON, ANDESCON 2020*. Institute of Electrical and Electronics Engineers Inc., Oct. 2020. ISBN: 9781728193656. DOI: 10.1109/ANDESCON50619.2020.9272097.

[74] E. B. Öksüz and G. Çağdaş. "An assessment method for a designerly way of computational thinking". In: *A/Z ITU Journal of the Faculty of Architecture* 17.2 (2020), pp. 199–208. ISSN: 25647474. DOI: 10.5505/itujfa.2020.86729.

[75] T. H. Park, M. C. Kim, S. Chhabra, B. Lee, and A. Forte. "Reading hierarchies in code: Assessment of a basic computational skill". In: *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*. Vol. 11-13-July. Association for Computing Machinery, July 2016, pp. 302–307. ISBN: 9781450342315. DOI: 10.1145/2899415.2899435. URL: http://dx.doi.org/10.1145/2899415.2899435.

[76] L. Pollock, C. Mouza, K. R. Guidry, and K. Pusecker. "Infusing computational thinking across disciplines: Reflections & lessons learned". In: *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. Vol. 7. Association for Computing Machinery, Inc, Feb. 2019, pp. 435–441. ISBN: 9781450358903. DOI: 10.1145/3287324.3287469. URL: https://doi.org/10.1145/3287324.3287469.

8

[77] S. M. Pulimood, K. Pearson, and D. C. Bates. “A study on the impact of multidisciplinary collaboration on computational thinking”. In: *SIGCSE 2016 - Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. Association for Computing Machinery, Inc, Feb. 2016, pp. 30–35. ISBN: 9781450338561. DOI: 10.1145/2839509.2844636. URL: http://dx.doi.org/10.1145/2839509.2844636.

[78] M. M. Rahman and R. Paudel. “Visual programming and interactive learning based dynamic instructional approaches to teach an introductory programming course”. In: *Proceedings - Frontiers in Education Conference, FIE*. Vol. 2018-Octob. Institute of Electrical and Electronics Engineers Inc., Mar. 2019. ISBN: 9781538611739. DOI: 10.1109/FIE.2018.8658581.

[79] R. Ravikumar and P. Sadhwani. “Utilizing a data analytics project to increase student awareness of contemporary global issues”. In: *2020 Advances in Science and Engineering Technology International Conferences, ASET 2020*. Institute of Electrical and Electronics Engineers Inc., Feb. 2020. ISBN: 9781728146409. DOI: 10.1109/ASET48392.2020.9118395.

[80] A. Rojas-López and F. J. García-Pealvo. “Interpretation of computational thinking evaluation results for enrollment prediction”. In: *ACM International Conference Proceeding Series*. New York, NY, USA: ACM, 2019, pp. 13–18. ISBN: 9781450371919. DOI: 10.1145/3362789.3362803. URL: https://doi.org/10.1145/3362789.3362803.

[81] M. Romero, A. Lepage, and B. Lille. “Computational thinking development through creative programming in higher education”. In: *International Journal of Educational Technology in Higher Education* 14.1 (Dec. 2017), pp. 1–15. ISSN: 23659440. DOI: 10.1186/s41239-017-0080-z. URL: https://educationaltechnologyjournal.springeropen.com/articles/10.1186/s41239-017-0080-z.

[82] N. Senske. “Evaluation and impact of a required computational thinking course for architecture students”. In: *Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE*. New York, NY, USA: ACM, 2017, pp. 525–530. ISBN: 9781450346986. DOI: 10.1145/3017680.3017750. URL: http://dx.doi.org/10.1145/3017680.3017750.

[83] A. Settle. “Computational thinking in a game design course”. In: *SIGITE’11 - Proceedings of the 2011 ACM Special Interest Group for Information Technology Education Conference*. New York, New York, USA: ACM Press, 2011, pp. 61–66. ISBN: 9781450310178. DOI: 10.1145/2047594.2047612.

[84] D. E. Sondakh, K. Osman, and S. Zainudin. “Holistic assessment of computational thinking for undergraduate: Reliability and convergent validity”. In: *ACM International Conference Proceeding Series*. New York, NY, USA: ACM, 2019, pp. 241–245. ISBN: 9781450372541. DOI: 10.1145/3369255.3369303. URL: https://doi.org/10.1145/3369255.3369303.

[85] D. E. Sondakh, K. Osman, and S. Zainudin. "A Pilot Study of an Instrument to Assess Undergraduates' Computational thinking Proficiency". In: *International Journal of Advanced Computer Science and Applications* 11.11 (2020), pp. 263–273. ISSN: 21565570. DOI: 10.14569/IJACSA.2020.0111134. URL: www.ijacsa.thesai.org.

[86] K. Takemata, A. Minamide, and A. Kodaka. "Development for an introductory active learning program: Utilize a short-term design project". In: *EDULEARN19 Proceedings*. Vol. 1. 2019, pp. 2673–2678. DOI: 10.21125/edulearn.2019.0727.

[87] S. Thangavelu, V. Rao, C. K. Shyamala, and C. S. Velayutham. "Introductory programming using non-textual modalities - An empirical study on skill assessment using rainfall problem". In: *IEEE Global Engineering Education Conference, EDUCON*. Vol. April-2019. IEEE Computer Society, Apr. 2019, pp. 867–871. ISBN: 9781538695067. DOI: 10.1109/EDUCON.2019.8725122.

[88] Y. Wang, C. Ault, T. Nakra, A. Salgian, and M. Stone. "An interdisciplinary undergraduate course bridging the gaps between engineering, science, and the arts". In: *ASEE Annual Conference and Exposition, Conference Proceedings*. American Society for Engineering Education, 2010. DOI: 10.18260/1-2--15823.

[89] L. V. Atieno, R. K. A. Baafi, and T. S. Marta. "Are computational thinking skills measurable? An analysis". In: *CEUR Workshop Proceedings*. Vol. 2650. 2020, pp. 12–23. URL: http://ceur-ws.org.

[90] R. Libeskind-Hadas and E. Bush. "A first course in computing with applications to biology". In: *Briefings in Bioinformatics* 14.5 (Sept. 2013), pp. 610–617. ISSN: 14774054. DOI: 10.1093/bib/bbt005. URL: https://academic.oup.com/bib/article/14/5/610/218260.

8

[91] C. K. Shyamala, C. S. Velayutham, and L. Parameswaran. "Teaching computational thinking to entry-level undergraduate engineering students at Amrita University". In: *IEEE Global Engineering Education Conference, EDUCON*. IEEE Computer Society, June 2017, pp. 1731–1734. ISBN: 9781509054671. DOI: 10.1109/EDUCON.2017.7943084.

[92] J. Wing. "Research Notebook: Computational Thinking-What and Why". In: *The Link/Carneige Mellon* (2011).

[93] ABET Computing Accreditation Commission. *Criteria for Accrediting Computing Programs: Effective for Reviews During the 2014-2015 Accreditation Cycle*. 2013. URL: http://www.abet.org/wp-content/uploads/2015/04/C001-14-15-CAC-Criteria.pdf%5Cnhttp://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2015-2016/.

[94] P. E. Newton. "Clarifying the purposes of educational assessment". In: *Assessment in education* 14.2 (2007), pp. 149–170.

[95] M. Valle Torre, C. Oertel, and M. Specht. “The Sequence Matters in Learning-A Systematic Literature Review”. In: *Proceedings of the 14th Learning Analytics and Knowledge Conference*. 2024, pp. 263–272.

[96] X. Zhang, M. Specht, and M. Valle Torre. “An investigation on integration of computational thinking into engineering curriculum at delft university of technology”. In: *Towards a new future in engineering education, new scenarios that european alliances of tech universities open up*. Universitat Politècnica de Catalunya. 2022, pp. 890–901.

[97] B. E. Seely. “Patterns in the history of engineering education reform: A brief essay”. In: *Educating the engineer of 2020: Adapting engineering education to the new century* (2005), pp. 114–130.

[98] S. A. Papert. *Mindstorms: Children, computers, and powerful ideas*. Basic books, 2020.

[99] X. Zhang, E. Aivaloglou, and M. Specht. “Teachers’ Intention to Integrate Computational Thinking Skills in Higher Education: A Survey Study in the Netherlands”. In: *2024 IEEE Global Engineering Education Conference (EDUCON)*. IEEE. 2024, pp. 1–9.

[100] S. Grover and R. Pea. “Computational Thinking in K-12: A Review of the State of the Field”. In: *Educational Researcher* 42.1 (2013), pp. 38–43. ISSN: 0013189X. DOI: 10.3102/0013189X12463051.

[101] A. Yadav, N. Zhou, C. Mayfield, S. Hambrusch, and J. T. Korb. “Introducing computational thinking in education courses”. In: *SIGCSE’11 - Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*. 2011, pp. 465–470. ISBN: 9781450305006. DOI: 10.1145/1953163.1953297. URL: http://cs4edu.cs.purdue.edu/comp_think.

[102] A. Yadav, C. Mayfield, N. Zhou, S. Hambrusch, and J. T. Korb. “Computational Thinking in Elementary and Secondary Teacher Education”. In: *ACM Transactions on Computing Education (TOCE)* 14.1 (Mar. 2014). ISSN: 19466226. DOI: 10.1145/2576872. URL: https://dl.acm.org/doi/10.1145/2576872.

[103] J. Cuny. “Transforming computer science education in high schools”. In: *Computer* 44.6 (2011), pp. 107–109. ISSN: 00189162. DOI: 10.1109/MC.2011.191. URL: http://csta.acm.org/Research/.

[104] S. Yee Lye and J. Hwee Ling Koh. “Review on teaching and learning of computational thinking through programming: What is next for K-12?” In: (2014). DOI: 10.1016/j.chb.2014.09.012. URL: http://dx.doi.org/10.1016/j.chb.2014.09.012.

[105] F. D. Davis. “Perceived usefulness, perceived ease of use, and user acceptance of information technology”. In: *MIS quarterly* (1989), pp. 319–340.

[106] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis. “User acceptance of information technology: Toward a unified view”. In: *MIS quarterly* (2003), pp. 425–478.

8

[107] U. L. Ling, T. C. Saibin, J. Labadin, and N. A. Aziz. "Preliminary investigation: Teachers' perception on computational thinking concepts". In: *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)* 9.2-9 (2017), pp. 23–29.

[108] O. Durodolu. "Technology Acceptance Model as a predictor of using information system'to acquire information literacy skills." In: *Library Philosophy & Practice* (2016).

[109] G. Fessakis and S. Prantsoudi. "Computer Science Teachers' Perceptions, Beliefs and Attitudes on Computational Thinking in Greece." In: *Informatics in Education* 18.2 (2019), pp. 227–258.

[110] B. Bruggink. "Behavioural Intention of Teachers for Teaching Computational Thinking in Primary Education". MA thesis. 2019.

[111] M. Specht and R. J. Joosse. "Technology Acceptance and Teacher Attitude for Computational Thinking in the Netherlands". In: *CoolThink@ JC* (2019), p. 113.

[112] Y. B. Kafai and Q. Burke. "Computer programming goes back to school". In: *Phi Delta Kappan* 95.1 (2013), pp. 61–65.

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

[114] S. Einhorn. "Microworlds, computational thinking, and 21st century learning". In: *LCSI White Paper* 2 (2012), pp. 2–10.

[115] ISTE and CSTA. *leadership toolkit first edition*. Tech. rep. [Accessed 17-08-2024]. 2011. URL: https://cdn.iste.org/www-root/2020-10/ISTE_CT_Leadership_Toolkit_booklet.pdf?_ga=2.241613378.1334721248.1633275757-262716800.1628547941.

8

[116] D. Weintrop, E. Beheshti, M. Horn, K. Orton, K. Jona, L. Trouille, and U. Wilensky. "Defining computational thinking for mathematics and science classrooms". In: *Journal of science education and technology* 25 (2016), pp. 127–147.

[117] S. Grover and R. Pea. "Computational thinking: A competency whose time has come". In: *Computer science education: Perspectives on teaching and learning in school* 19.1 (2018), pp. 19–38.

[118] I. Ajzen. "The theory of planned behavior". In: *Organizational behavior and human decision processes* 50.2 (1991), pp. 179–211.

[119] E. M. Rogers. *Diffusion of innovations*. Simon and Schuster, 2010.

[120] J. L. Hale, B. J. Householder, and K. L. Greene. "The theory of reasoned action". In: *The persuasion handbook: Developments in theory and practice* 14.2002 (2002), pp. 259–286.

[121] R. L. Thompson, C. A. Higgins, and J. M. Howell. "Influence of experience on personal computer utilization: Testing a conceptual model". In: *Journal of management information systems* 11.1 (1994), pp. 167–187.

[122] M. Igbaria, S. Parasuraman, and J. J. Baroudi. "A motivational model of microcomputer usage". In: *Journal of management information systems* 13.1 (1996), pp. 127–143.

[123] D. H. Schunk. *Social cognitive theory.* American Psychological Association, 2012.

[124] J. Bethlehem. *Applied survey methods: A statistical perspective.* John Wiley & Sons, 2009.

[125] R. Glaser. "Education and thinking: The role of knowledge." In: *American psychologist* 39.2 (1984), p. 93.

[126] M. Bower and K. Falkner. "Computational Thinking, the Notional Machine, Pre-service Teachers, and Research Opportunities." In: *ACE* 160 (2015), pp. 37–46.

[127] M. Armoni. "On teaching abstraction in CS to novices". In: *Journal of Computers in Mathematics and Science Teaching* 32.3 (2013), pp. 265–284.

[128] L. Nakar, M. Friebroon, and M. Armoni. "From Modelling to Assessing Algorithmic Abstraction–the Missing Dimension". In: *Proceedings of the 23rd Koli Calling International Conference on Computing Education Research.* 2023, pp. 1–12.

[129] D. Aharoni. "Cogito, Ergo sum! cognitive processes of students dealing with data structures". In: *Proceedings of the thirty-first SIGCSE technical symposium on Computer science education.* 2000, pp. 26–30.

[130] Q. Cutts, S. Esper, M. Fecho, S. R. Foster, and B. Simon. "The abstraction transition taxonomy: Developing desired learning outcomes through the lens of situated cognition". In: *Proceedings of the ninth annual international conference on International computing education research.* 2012, pp. 63–70.

[131] O. Hazzan. "Reducing abstraction level when learning abstract algebra concepts". In: *Educational Studies in Mathematics* 40 (1999), pp. 71–90.

[132] M. Friebroon-Yesharim, R. Ben-Bassat Levy, and M. Armoni. "Algorithmic Abstraction in Computer Science Curricula for Primary Schools: The Case of a National Curriculum for 4th Grade". In: *Proceedings of the 2023 Conference on United Kingdom & Ireland Computing Education Research.* 2023, pp. 1–7.

[133] J. Hartmanis. "Turing award lecture on computational complexity and the nature of computer science". In: *Communications of the ACM* 37.10 (1994), pp. 37–43.

[134] D. Statter and M. Armoni. "Teaching abstraction in computer science to 7th grade students". In: *ACM Transactions on Computing Education (TOCE)* 20.1 (2020), pp. 1–37.

[135] M. E. Kiger and L. Varpio. "Thematic analysis of qualitative data: AMEE Guide No. 131". In: *Medical teacher* 42.8 (2020), pp. 846–854.

A

Questionnaire on Teachers' Understanding of CT Skills and Perceptions on Integrating it

Welcome

Computational Thinking is thinking process that allows one to formulate a solution to a problem to be executed by any information processing agency (being either human or electronic machine).

This survey is **conducted by the Delft University of Technology**. The **aim of the research** is to gain insight into teachers' perceptions and views on both Computational Thinking (CT) skills and intention of integrating them into teaching. That means, you will answer questions on your understanding and attitudes towards CT skills and integration of CT into teaching and learning practices.

Your answers will be gathered in a manner that each individual is unrecognizable. The results **will be reported** in national and international scientific publications, international conferences and in the media.

Filling out the survey will take **approximately <u>10</u> minutes** of your time. Your **answers will be treated confidentially**. The results will be aggregated and will not be traceable to individuals or schools.

Thanks for your participation and cooperation. If you have any questions about this research, **please contact** X.Zhang-14@tudelft.nl

* To get the best user experience for filling out the survey, Chrome browser is recommended. *

Consent

- I give permission for the use of the data (the answers for the questions) collected in this questionnaire for scientific research.
- I have read the information regarding this research and I have had the opportunity to ask the researcher additional questions if there are any uncertainties.
- I understand that any information I provide in relation to this research will be collected anonymously and cannot be traced back to me.
- I understand that I may discontinue this questionnaire at any time without giving any reason.

If you have read the above points and agree to participate in the study.

please digitally sign the consent form by entering the information asked below.

Signature
I agree to participate in this research.

<u>*To confirm that you participate in this study **voluntarily** and make sure we can **track back to your answers without your real name and ID** (if you want to discontinue with the questionnaire*</u>, a code will be generated by you according to a rule given below). Therefore,

please note it down and keep it somewhere until you are sure you will not discontinue this research.

We kindly ask you to enter a code that align with the following rule:

Rule: the **date** today + the **birthday** of **your favorite person +** the **capitalized first letters** of **your favorite person's first name and last name.**

For example**:** for 2022 October 24th + 1991 November 21th + Karson Mars the code would be **202210241991 1121MK**.

____

Greetings Hi there!

It is so great to have you as a participant of this survey study.

We hope you enjoy filling out the survey and find it helpful also for your sake to reflect on your own thoughts.

Let's get started!

Q1 I am a:

- ◯ Male (1)
- ◯ Female (2)
- ◯ Non-binary / third gender (3)
- ◯ Prefer not to say (4)

Q2 My age is in the range of:

- ◯ 20 or below 20 (1)
- ◯ 21-30 (2)
- ◯ 31-40 (3)
- ◯ 41-50 (4)
- ◯ 51-60 (5)
- ◯ Above 60 (6)

Q3 My highest academic degree is:

- ○ Bachelor of Science (BSc) (1)
- ○ Bachelor of Arts (BA) (2)
- ○ Master of Science (MSc) (3)
- ○ Master of Arts (MA) (4)
- ○ PhD (5)
- ○ Others (6) ____

Q4 I have been teaching in higher education (including university, college) for

- ○ 1-3 year (s) (1)
- ○ 3-5 years (2)
- ○ 5-10 years (3)
- ○ More than 10 years (4)

Q5 I have mainly taught topics / areas / domains / majors of (in higher education)...
(If there is more than one, please list them and use a comma to separate different items, **e.g. art, civil engineering, statistics, finance.**

____

____

Q6 The following statements match my experience with Computer Science (CS) and / or Programming and Computational Thinking (CT):

	Never (1)	Once (2)	Sometimes (3)	Repeatedly (4)	Regularly (5)
I have attended workshops / trainings related to computer science and / or progamming (1)	○	○	○	○	○
I have attended workshops / trainings related to Computational Thinking . (2)	○	○	○	○	○

Q7 In my understanding, the relation between Computational Thinking (CT) and Computer Science (CS) is

- ◯ CT is a concept wider than CS, because it further includes the ability of solving problems in various disciplines, even without the use of computers (1)
- ◯ CT and CS have common attributes, but each one also has special, discrete attributes (2)
- ◯ CS is a concept wider than CT, because it further includes, e.g. the study of computation, programming languages, and computer hardware (3)
- ◯ CS and CT are the same (4)

Q8 I consider the following Computational Thinking (CT) dimensions, in my domain(s) as:

	not important (1)	slightly important (2)	fairly important (3)	important (4)	very important (5)
Abstraction (1)	◯	◯	◯	◯	◯
Algorithmic Thinking (2)	◯	◯	◯	◯	◯
Automation (3)	◯	◯	◯	◯	◯
Being Incremental and Iterative (4)	◯	◯	◯	◯	◯
Collaboration and Cooperation (5)	◯	◯	◯	◯	◯
Communication (6)	◯	◯	◯	◯	◯
Creativity and Creation (7)	◯	◯	◯	◯	◯
Critical Thinking (8)	◯	◯	◯	◯	◯
Data (9)	◯	◯	◯	◯	◯
Evaluation (10)	◯	◯	◯	◯	◯

Expressing and Questioning (11)	o	o	o	o	o
Generalization (12)	o	o	o	o	o
Information Processing (13)	o	o	o	o	o
Logic and Logical Thinking (14)	o	o	o	o	o
Modularization / Modelling (15)	o	o	o	o	o
Organization (16)	o	o	o	o	o
Pattern Recognition (17)	o	o	o	o	o
Planning (18)	o	o	o	o	o
Problem Decomposition (19)	o	o	o	o	o
Problem Solving (20)	o	o	o	o	o
Reasoning (21)	o	o	o	o	o
Reflection (22)	o	o	o	o	o
Self-Efficacy, Self-Competency, and Confidence (23)	o	o	o	o	o
Synchronization (24)	o	o	o	o	o
Testing and Debugging (25)	o	o	o	o	o

User Interactivity (26)	○	○	○	○	○

Q9 In my opinion, Computational Thinking (CT) can be taught by:

- ○ Teachers with Computer Science (CS) education (1)
- ○ Teachers with proper training on Computer Science (CS) knowledge (2)
- ○ All teachers, regardless of Computer Science (CS) education experience (3)

Q10 I think it is proper to integrate CT into education at (It is **possible** to choose **more than one answer**.)

- ▢ Tertiary education / Higher education (1)
- ▢ High school / Secondary education (2)
- ▢ Primary school / Primary education (3)

Q11 For the following statements, please indicate if you agree or disagree:

	Strongly Disagree (1)	Somewhat disagree (2)	Neither agree nor disagree (3)	Somewhat agree (4)	Strongly agree (5)
I would find integrating CT into teaching useful in my job (1)	○	○	○	○	○
Integrating CT into teaching enables me to accomplish tasks more quickly (2)	○	○	○	○	○
Integrating CT into teaching increases my productivity (3)	○	○	○	○	○
Integrating CT in teaching increases my chances of getting a raise in salary / employment opportunities (4)	○	○	○	○	○
The way of integrating CT into teaching would be clear and understandable (5)	○	○	○	○	○

It would be easy for me to become skillful at integrating CT into teaching (6)	○	○	○	○	○
I would find integrating CT into teaching easy to implement (7)	○	○	○	○	○
Learning to integrate CT into teaching is easy for me (8)	○	○	○	○	○
Integrating CT into teaching is a good idea (9)	○	○	○	○	○
Integrating CT into teaching makes work more interesting (10)	○	○	○	○	○
Integrating CT into teaching is fun (11)	○	○	○	○	○
I would like to do teaching where CT is integrated (12)	○	○	○	○	○
People who influence my behavior think that I should integrate CT into teaching (13)	○	○	○	○	○
People who are important to me think that I should integrate CT into teaching (14)	○	○	○	○	○
The senior management of this school / department has been helpful in the integration of CT into teaching (15)	○	○	○	○	○
In general, the organization has supported in the integration of CT into teaching (16)	○	○	○	○	○
I have the knowledge necessary to integrate CT into teaching (17)	○	○	○	○	○

I have the resources necessary to integrate CT into teaching (18)	○	○	○	○	○
Integrating CT into teaching is conflicting with other educational plans of the organization (19)	○	○	○	○	○
A specific person (or group) is available for assistance with difficulties of integrating CT into teaching (20)	○	○	○	○	○
Although it might be helpful, integrating CT into teaching would certainly not be compulsory in my job (21)	○	○	○	○	○
My boss would not require me to integrate CT into teaching (22)	○	○	○	○	○
My superiors would not expect me to integrate CT into teaching (23)	○	○	○	○	○
Integrating CT into teaching would be voluntary (as opposed to required by superiors / job) (24)	○	○	○	○	○
I intend to integrate CT into teaching in the near future (25)	○	○	○	○	○
I predict I would integrate CT into teaching in the near future (26)	○	○	○	○	○
I plan to integrate CT into teaching in the near future (27)	○	○	○	○	○

Q12 **Which skills** related to **computer science**, **data science**, or **machine learning** do you think **are important** for the graduates **in your domain / area / topic**?

____

____

Q13 If you are currently teaching outside of Netherlands, can you please indicate which country that is?

____

B

Experiment Material for Measuring Algorithmic Abstraction Skills

Welcome to this survey!

Computational Thinking skills are vital for problem-solving. Algorithms and abstraction are essential components of computational thinking. By participating in this study, you will be applying computational thinking to solve problems.

This study is conducted by the Delft University of Technology. The research aims to learn what level of algorithmic abstraction skills people have and need in different domains. The survey includes questions about your age, major, as well as questions about algorithmic abstraction and problem-solving methods on different levels of difficulty.

Your answers will be gathered in a manner in which each individual is unrecognizable. The answer will have no impact on your study/grades, it will be only used for research purposes.The results will be shared in national and international scientific publications and international conferences.

Participation in this study will take **an hour of your time**. You will be **rewarded a voucher of value 10 EURO** for participating in this study. If you have questions, please contact the researcher at <X.Zhang-14@tudelft.nl>

Consent

- I give permission for the use of the data collected during this research for scientific research.
- I give permission for the de-identified data that I provide to be archived in Project Storage at TU Delft and Git(lab)/subversion repository at TU Delft so it can be used for future research and learning.
- I have read the information regarding this research and I have had the opportunity to ask the researcher additional questions if there are any uncertainties.
- I understand that I will be compensated for my participation by receiving a gift card with a value of 10 euros.
- I understand that any information I provide in relation to this research will be collected anonymously and cannot be traced back to me.
- I understand that I may discontinue this research at any time without giving a reason.
- If you have read the above points and agree to participate in the study, please digitally sign the consent form below by entering the information asked below.

Signature
I agree to participate in this research. **Enter the participant ID from the SONA system or the researcher in the cell below**. This code will be used if you want to discontinue this research, therefore please write it down and keep it somewhere until you are sure you will not discontinue this research.

____

In this part, you'll share some basic details about yourself.

B-Q1 **I identify myself as:**

- ◯ Male (1)
- ◯ Female (2)
- ◯ Non-binary / third gender (3)
- ◯ Prefer not to say (4)

B-Q2 **I am between the ages of:**

- ◯ 20 or below 20 (1)
- ◯ 21-30 (2)
- ◯ 31-40 (3)
- ◯ 41-50 (4)
- ◯ 51-60 (5)
- ◯ Above 60 (6)

B-Q3 **I am currently enrolled in a program for:**

- ◯ Bachelor of Science (BSc) (1)
- ◯ Bachelor of Arts (BA) (2)
- ◯ Master of Science (MSc) (3)
- ◯ Master of Arts (MA) (4)
- ◯ PhD (5)
- ◯ Other, please specify: (6)

B-Q4 **I am in the:**

- ◯ 1st year of study (1)
- ◯ 2nd year of study (2)
- ◯ 3rd year of study (3)
- ◯ 4th year of study (4)
- ◯ Other, please specify: (5) ____

B-Q5 **I am in the field of:**

- ◯ Science, Technology, Engineering, Mathematics (STEM) studies (1)
- ◯ Non-STEM studies (e.g. History, Arts, Music) (2)

B-Q6 **My current area of study is:**

- ◯ Computer Science (1)
- ◯ Non Computer Science (2) ____
- ◯ Other, please specify (3) ____

B-Q7 **The following statement match my experience:**

	Never (1)	Once (2)	Sometimes (3)	Repeatedly (4)	Regularly (5)
I have taken or am currently taking courses /workshops/training in computer science, programming, Computational Thinking, or algorithmic problem-solving. (1)	◯	◯	◯	◯	◯

Display This Question:

If The following statement match my experience: != Never

B-Q7a **Please tell us about the computer science, programming, or algorithmic problem-solving courses/workshops/training you've taken or are currently taking.**

____

____

____

____

____

B-Q8 Regarding programming, my experience with the following programming language can be identified as:

	No level of competence (1)	Low level of competence (2)	Average level of competence (3)	Moderately high level of competence (4)	High Level of competence (5)
Python (1)	o	o	o	o	o
C (2)	o	o	o	o	o
C++ (3)	o	o	o	o	o
Matlab (4)	o	o	o	o	o
Java (5)	o	o	o	o	o
R (7)	o	o	o	o	o
Other, please specify: (6)	o	o	o	o	o

Next, you'll read a story about Lisa's day and assist her with some issues. The story has four parts, and your task is to use your knowledge to help Lisa as best as you can.

LISA's Day - Part 1

Lisa recently started working as a library assistant, and her friend Lee is curious about her job. Lisa shared that her main tasks involve digitizing the library catalog and assisting with organizing activities. Lee asked for more details about Lisa's daily work, and Lisa explained:

"I begin my day with coffee and review the to-do list, usually messy, in an Excel sheet. To streamline, I categorize tasks into very urgent, urgent, and not-that-urgent, then rearrange them by priority.

Currently, I'm digitizing the library services, dealing with over 5,000 books listed in the Excel sheet arranged by topics. The current list is with some missing information for some books, and it is only possible to look up books by topics. To make it more efficient to look up books and insert new books. I am focusing on:

- Completing missing information for existing books.
- Planning an efficient arrangement of books, alphabetically or numerically, such as author or editorial.
- Implementing system functions for lending, returning, exchanging, adding, and removing books.
- I spend much of my day thinking about ways to optimize these tasks."

SP1-Q1 In Lisa's day, sorting can make her work more efficient. **Can you point out where and how sorting can be used in Lisa's day? Please select the sentences where sorting activities are mentioned in the text.**

____

____

SP1-Q2 **Please explain what sorting is, and give an example of how it can be done.**

____

____

SP1-Q3 **Please provide an example from your everyday life or work where sorting is helpful?**

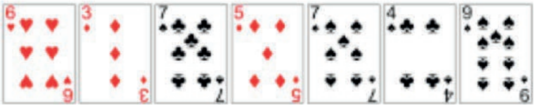
____

____

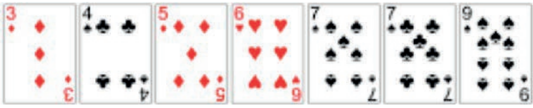
LISA's Day - Part 2

After listening to Lisa, Lee became intrigued by the method used to efficiently organize books by names, authors, publisher, or year of publication. Lee asked Lisa to explain the design of the solutions further, and Lisa compared it to sorting a deck of cards.

Cards unsorted:



Cards sorted:

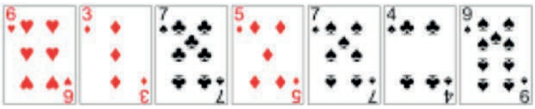


SP2-Q1 **Please provide your solution to get the unsorted cards sorted.**

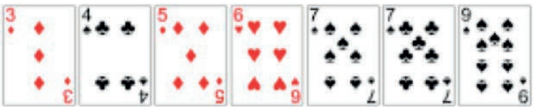
____

____

Let's make this more specific.
You start with the card order:



it should end with the following order:



SP2-Q2 **Please describe the operation that you take for each step to get the cards sorted, and show the results of each round using the solution you described for the previous question.**

____

____

SP2-Q3 **Can the way you sort a deck of cards be used to efficiently organize over 5,000 books? If not, suggest a strategy you think would be suitable for efficiently sorting such a large number of books.**

- ◯ Yes (1)
- ◯ No (2) ____

SP2-Q4 **Please provide a step-by-step guide to explain to Lee how to organize books in A-Z order?**

____

____

Pseudo-code is a simple way to explain algorithms and software programs. This is an initial step in learning programming, as it provides a more organized and formal description of the steps in an algorithm.

Two examples of pseudo code for finding the maximum number between two numbers.

Example 1:

```
Function findMaximum(num1, num2):
  if num1 > num2:
    return num1
  else:
    return num2
```

Example 2:

```
Start with two numbers, let's call them num1 and num2.

If num1 is greater than num2, then:
  The maximum is num1.
Else:
  The maximum is num2.

Print or return the maximum value.
```

SP2-Q5 **Please create a pseudo-code for the sorting process you explained in answer to the last question (SP2-Q4)?**

____

____

After Lisa explained sorting in simple terms (organizing books in A-Z order), Lee asked about other strategies or designs for solving the problem. Lisa shared a list of strategies she learned online for sorting a list of items.

SP2-Q6 Please specify which strategies from the list below you recognize. Also, provide an example scenario where that strategy can be applied.

- ☐ A. Swap adjacent elements if they are in the wrong order. (1) ____
- ☐ B. Divide items into smaller lists, sort each, and merge them for the final sorted list. (2) ____
- ☐ C. Choose an element as a pivot, arrange the list around it for the correct position. (3) ____
- ☐ D. Select the smallest (or largest) element from the unsorted part and move it to the sorted part. (4) ____
- ☐ E. Similar to sorting playing cards in your hands, virtually splitting the list into sorted and unsorted parts, and placing values in the correct positions. (5) ____

LISA's Day - Part 3

Meanwhile, Lee discovered that Lisa needed to use Python programming to apply the strategy for organizing books alphabetically. Curious to see how it's done, Lisa demonstrated what a program solution might look like for sorting a list of books in A-Z order in Python.
Note: The book names are stored as a list of names.

Accidentally, Lee found a potential Python solution for sorting a deck of cards. The solution is given as a list of statements. Lisa looked at the statements and rearranged them to create a functional Python solution.

SP3-Q1 **Can you arrange them in the correct order? (Drag and drop the statements to the correct order)**

```
______ def sort_books(book_list): (1)
______         j = i - 1 (2)
______     return book_list (3)
______         while j >= 0 and current_book < book_list[j]: (4)
______     for i in range(1, len(book_list)): (5)
______             book_list[j + 1] = book_list[j] (6)
______         current_book = book_list[i] (7)
______         book_list[j + 1] = current_book (8)
______           j -= 1 (9)
```

SP3-Q2 **Write a basic Python algorithm to organize books in alphabetical order (from A – Z) using the strategy you provided in SP2-Q5.**

(Please use the function template provided below)

```
def sort_books(book_list):

    # Here is the  code python algorithm

    return book_list
```

____

____

LISA's Day - Part 4
Lisa generated solutions quickly, but unfortunately, her solution had some errors.
SP4-Q1 **Can you help find problems in the code and fix them?**

```
def sort_books(book_list):
    for i in range(1, len(book_list) + 1):
```

```
        current_book = book_list[i]
        j = i

        while j > 0 and current_book < book_list[j - 1]:
            book_list[j + 1] = book_list[j - 1]
            j -= 1

        book_list[j + 1] = current_book

    return book
```

What is the problem of the code above?

____

____

SP4-Q2 **Can you help find problems in the code and fix them?**

```
def sort_books(book_list):
    for i in range(len(book_list)):
        current_book = book_list[i]
        i -= 1

        while i > 0 and current_book > book_list[i - 1]:
            book_list[i] = book_list[i - 1] + 1
            i -= 1

        book_list = current_book

    return book_list
```

What is the problem of the code above?

____

____

SP4-Q3 **Please suggest Lisa some strategies to identify her errors in the code above.**

____

____

R - Q1 For tasks from Part 1 to Part 4 of LISA's day, which part includes the most challenging task(s) for you? Please share your thoughts on potential causes for you to make that choice.

- ▢ Part 1 Recognize and define sorting in a scenario. (1) ____
- ▢ Part 2 Design a sorting plan based on a scenario. (2) ____
- ▢ Part 3 Implement the sorting plan in a specific programming language (3) ____
- ▢ Part 4 Rectify the errors in the implementation of the sorting plan in a specific language. (4) ____
- ▢ None of the above (5) ____

In algorithmic abstraction problem-solving, different levels require understanding and solving problems based on the skills needed. Here are the definitions of the levels.

P-Q1 **Which levels of algorithmic abstraction have you already mastered?**

- ☐ A. Problem Level: I understand and represent input-output examples based on a given problem description. (1)
- ☐ B. Algorithm Level: I comprehend the problem, devise a solution, and design a step-by-step plan using clear components, executable by computers or humans. (2)
- ☐ C. Program Level: I understand the problem, create a solution design, and transform it into a written program in a specific programming language. (3)
- ☐ D. Execution Level: I understand the problem, create a solution design, transform it into a written program, review the solution by executing the program, and learn from the process. (4)
- ☐ None of the above. (5)

P-Q2 **Which levels of algorithmic abstraction do you think are essential to master in your domain?**

- ☐ A. Problem Level: Understand and represent input-output examples based on a given problem description. (1)
- ☐ B. Algorithm Level: Comprehend the problem, devise a solution, and design a step-by-step plan using clear components, executable by computers or humans. (2)
- ☐ C. Program Level: Understand the problem, create a solution design, and transform it into a written program in a specific programming language. (3)
- ☐ D. Execution Level: Understand the problem, create a solution design, transform it into a written program, review the solution by executing the program, and learn from the process. (4)
- ☐ None of the above. (5)

List of Publications

- 📘 **Zhang, X., Aivaloglou, F., and Specht, M. (2024).** A Systematic Umbrella Review on Computational Thinking Assessment in Higher Education. European Journal of STEM Education, 9(1), 02.

- 📘 **Zhang, X., Valle Torre, M., and Specht, M. (2022).** An Investigation on Integration of Computational Thinking into Engineering Curriculum at Delft University of Technology. In SEFI 2022 - 50th Annual Conference of the European Society for Engineering Education, Proceedings, 890–901. Universitat Politècnica de Catalunya. https://doi.org/10.5821

- 📘 **Zhang, X., Aivaloglou, F., and Specht, M. (2024).** Teachers‘ Intention to Integrate Computational Thinking Skills in Higher Education: A Survey Study in the Netherlands. 2024 IEEE Global Engineering Education Conference (EDUCON).

- 📘 **Zhang, X., Aivaloglou, F., and Specht, M.** Assessment of Algorithmic Abstraction Skills in Higher Education: An Application of the PGK Framework. (Accepted by 2025 IEEE Global Engineering Education Conference (EDUCON)).

- **Zhang, X., Glahn, C., Fanchamps, L. J. A., Specht, M. M. (Eds.) (2022).** CTE-STEM 2022 | Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022. TU Delft. https://doi.org/10.34641/mg.37

- **Zhang, X., and Specht, M. (2022).** Towards a computer-assisted Computational Thinking (CT) assessment system in higher education. In: CEUR Workshop Proceedings, 3292, 12-21.

- **Zhang, X., and Specht, M. (2022).** A Review of Reviews on Computational Thinking Assessment in Higher Education. In Proceedings of Sixth APSCE International Conference on Computational Thinking and STEM Education 2022 (CTE-STEM), 98–103. https://doi.org/10.34641/CTESTEM.2022.472.

- **Cambaz, D. and Zhang, X. (2024).** Use of AI-driven Code Generation Models in Teaching and Learning Programming: a Systematic Literature Review. SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education. Association for Computing Machinery (ACM), p. 172-178 7 p. (SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education; vol. 1).

📘 Included in this thesis.

ACKNOWLEDGEMENTS

"No one can whistle a symphony. It takes an orchestra to play it." These words, attributed to Halford Luccock, a minister and professor of divinity at Yale, resonate deeply with me as I reflect on my PhD journey. Completing a PhD is much like orchestrating a symphony, and it would not have been possible without the support and collaboration of numerous individuals. I would like to express my profound gratitude to all those who have contributed to this endeavor.

Dear Marcus, your unwavering support and patience have been invaluable. Your willingness to give me space to explore, paired with your clear explanations of educational concepts and statistics, has greatly enhanced my understanding. I deeply appreciate your thorough review of my questionnaire and your collaborative spirit in brainstorming the design of the assessment instrument. Your ability to generate creative ideas throughout various stages of the research has been truly inspiring. Our meetings, often accompanied by refreshing jazz music, provided much-needed inspiration and clarity during challenging times.

Dear Felienne, your critical insights into my work on AI and your thought-provoking comments about post-PhD career prospects have left a lasting impression. Though I am still navigating the path you envisioned, the seeds of your ideas have been planted, and I look forward to their growth in the future. Your perspective has been immensely valuable, and for that, I am deeply grateful.

Dear Fenia, your personalized guidance and consistent support have been exceptional. From setting weekly goals and reviewing progress to advising on paper submissions and offering insights into networking and collaboration, you have played a crucial role in my development. Our discussions on topics such as managing emotions after rejection and navigating cross-cultural communication illuminated the path when it seemed to narrow and darken. Your suggestions have significantly shaped my work from its inception to completion.

I extend my heartfelt thanks to my graduation committee members, Prof. dr. M. M. Bonsangue, Prof. dr. A. van Deursen, Prof. dr. -Ing. U. Schroeder, Dr. E. Rahimi and Prof. dr. ir. G. J. P. M. Houben (reserve member). Your rigorous review of my work, combined with your invaluable expertise and mentorship, has been instrumental in finalizing my thesis. I hope to have the honor of collaborating with you in the future.

To my family and friends—especially my parents, my younger sister, and dear friends such as Wenyi, Jiaqin, Yujing, Cees, Minrui, Madalina, Anita, Jiayi, Yun, Daniel, Tianyi, Yuxuan, Shuaidong, Marijke, Nico, Koen, Ruben, Frank—words cannot express my gratitude for your love, patience, and belief in me. Your unwavering support has been the foundation that kept me grounded throughout this journey. I am deeply thankful for your encouragement and understanding.

A heartfelt thanks to my colleagues and friends at Delft University of Technology. Your camaraderie and encouragement made this challenging journey more manageable, and I will always cherish the many conversations, both academic and personal, that we shared. Special thanks to Ali, Bibeg, Erna, Frank, Gitte, Gillian, Gosia, Ioana, Esther, Marlies, Manuel, Merel, Nesse, Noortje, Shirong, Sylvia, Vivian, Xiaomei, Yoon—your support during the toughest moments has been invaluable.

I would also like to thank the administrative and technical staff at Delft University of Technology for your assistance and for creating a supportive environment that allowed me to focus on my research.

Lastly, I extend my gratitude to all those not specifically mentioned here but who have, in various ways, been part of this journey. Whether through brief conversations, words of encouragement, or simply sharing moments of understanding, your presence has left a lasting impact, and for that, I am deeply thankful.

Thank you all for being part of this symphony and contributing to the richness of this experience.

Curriculum Vitæ

Xiaoling Zhang

11-01-1995	Born in Chongqing, China.

Education

2013–2017	Undergraduate in Software Engineering University of Electronics Science and Technology in China Chengdu, China Thesis: Office Automation (OA) System Development
2017–2019	Master in Advanced Data Analytics and Computer Science Leiden University Leiden, Netherlands Thesis: Process Extraction from Scientific Literature
2020–2025	PhD. Educational Technology Delft University of Technology Delft, Netherlands Thesis: Computational Thinking Assessment in Higher Education Promotors: Prof. dr. M. M. Specht and Prof. dr. ir. F. F. J. Hermans Co-promotor: Dr. ir. E. Aivaloglou

Working Experience

2016-2017	Research and Development Assistant
2019-2020	Education Technical Assistant

TUDelft