Bachelor thesis
Computing Science

Radboud University

Incorporating Computational Thinking in Calculus lessons: a Characterisation of Algorithmic Thinking and Generalisation Skills

Author:
Bart Janssen
s4630270

First supervisor/assessor:
prof. dr. E. Barendsen (Erik)
erik.barendsen@ru.nl

Second supervisor/assessor:
dr. C. Chytas (Christos)
christos.chytas@ru.nl

August 31, 2021

Abstract

In the past decade, the importance of developing computational thinking (CT) and mathematical thinking (MT) skills in an increasingly specialised and complex society has seen increased attention. On behalf of the Netherlands Initiative for Education Research (NRO), a team of educational researchers and teachers in the field of mathematics and computer science made an effort to realize the integration of CT into calculus lessons. In this thesis, we tried to characterize the use of algorithmic thinking and generalisation skills, two elements of CT and MT, by 16-17 year old students after a 5-lesson calculus course that revolved around the use of the mathematical tool GeoGebra. Analysis of data from 20 students consisting of pre- and post-tests, student work in workbooks and GeoGebra, and mini-interviews demonstrated that students welcomed the use of GeoGebra and were adequate in solving tasks that required the use of algorithmic thinking and generalisation skills, such as constructing and evaluating algorithms, and the use of logical structures. Findings suggest that students needed time to familiarize themselves with the syntax and digital environment, while not encountering major challenges regarding algorithmic thinking and generalisation. The results also suggest that the majority of mistakes that were made during the course could be attributed to mathematical shortcomings rather than shortcomings related to CT.

Keywords: *algorithmic thinking, calculus, computational thinking, generalisation, GeoGebra*

Contents

List of Figures

List of Tables

Acknowledgements

I want to give my thanks to the Netherlands Initiative for Education Research (NRO) and all other persons involved in both the design, data collection and data analysis of this course. It has been a tough year for us all, and for me to have the opportunity to take part in this research was phenomenal. Special thanks goes out to my supervisor Christos, who helped me every step of the way, who always was available for questions, and who kept me sharp and motivated when times got tough. I could not have done this without his expertise and support throughout all stages of my research.

Chapter 1

Introduction

The importance of developing thinking skills that are essential to the 21^{st} century to prepare students for an increasingly specialized and complex society, has seen increased attention over the past decade. Popularized by Wing [53], computational thinking (CT) is a thinking process that can aid in this development. According to Wing [54], CT could be defined as: "*the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent*". Here we stress that such an information-processing agent can be either a human or a computer. Furthermore Wing argued that CT should be part of every 21^{st} century students' toolset and stressed that CT is not only limited to the field of computing science. Science, Technology, Engineering, Arts/Humanities and Mathematics (so-called STEAM education) also appears to benefit from the use of CT [6]. Unfortunately, integrating CT in STEAM education appears to be a bigger challenge than initially thought. CT is still an ill-defined construct, with differing definitions and no consensus on how to properly assess it [52]. However, despite these varying definitions, common elements of CT have been identified such as abstraction, algorithmic thinking, decomposition, generalisation, and pattern recognition [27, 41]. Another hindrance is that most educators think that teaching general problem solving is sufficient, which stems from a misunderstanding of the meaning of CT [48]. Furthermore, teachers' lack of training and unfamiliarity with computers or ICT also adds to a slowed integration [32].

In light of these difficulties, the Netherlands Initiative for Education Research (NRO) is conducting a multi-phase study that focuses on incorporating CT in K-12 mathematics education. A phase of this study was the design of a 5-lesson calculus course that focused on the use of algorithmic thinking and generalisation skills within the digital environment GeoGebra. The course has been rolled out in several schools throughout the Netherlands, and in this thesis we will try to provide a characterisation of how

students from one school that participated used these skills.

We conducted a data analysis on qualitative data from the participating school, which consisted of student pre- and post-test results, student work on the activities including workbooks and GeoGebra files, and recordings of interviews with students regarding the material and aspects of CT. Our findings will elucidate the learning outcomes and learning process of the students related to the use of algorithmic thinking, generalisation and the use of GeoGebra. This thesis attempts to function as a feed-forward in the design process of the research consortium, while also demonstrating how a tool such as GeoGebra can be used to foster aspects of CT and teach content related to calculus.

1.1 Outline

We will start this thesis in **Chapter 2** in which we will work towards defining our research questions by exploring the definitions of computational and mathematical thinking. Furthermore, we will investigate their interplay, assessment of computational thinking, and the mathematical tool GeoGebra. In **Chapter 3** we will elucidate the intervention design, and the process of data collection and analysis. Next, we will present the results of our analysis in **Chapter 4**. Thereafter, we will discuss and interpret these results in **Chapter 5** while answering the research questions of this thesis, identify limitations and give recommendations for future research. Lastly, in **Chapter 6** we will give a brief overview of our thesis and give some concluding remarks.

Chapter 2

Background

Within this chapter, we will shed some light on the varying definitions of CT and the concepts that can be associated with it. Additionally, we will take a closer look at mathematical thinking and how it relates to CT which will explain why CT can be integrated within mathematics education. Furthermore, we will highlight several methods that can be used when assessing CT skills and we will elaborate on how it relates to our research. Moreover, we will look at the digital tool GeoGebra and describe how it may stimulate students in using algorithmic thinking and generalisation skills. Lastly, we will define our research questions for this thesis.

2.1 Computational thinking

Ever since Wing [53] introduced the world with her definition of CT as a set of tools that *"involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science"*, research into the matter has seen increased attention. However, there still seems to be little agreement amongst scholars about a formal definition [22]. As a consequence, a variety of definitions emerged that according to Román-González, Pérez-González and Jiménez-Fernández [39] could be grouped into three categories: generic definitions, operational definitions and educational-curricular definitions.

The above-mentioned definition by Wing, as well as the definition given years later [54], are considered generic definitions. In an effort to come up with a more operational definition, the Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE) described a definition accompanied with several characteristics. They define CT as *"a problem-solving process that includes formulating problems in a way that enables us to use a computer and other tools to help solve them; logically organizing and analyzing data; representing data through abstractions such as models and simulations; automating solutions through*

algorithmic thinking (a series of ordered steps); identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources; generalizing and transferring this problem solving process to a wide variety of problems" [13].

In an attempt to disentangle all the varying definitions, Selby and Woollard [41] reviewed 22 articles concerning definitions of CT and consequently described CT as *"an activity, often product-oriented, associated with, but not limited to, problem-solving"*. They also provided some competencies that could be associated with CT such as abstracting, decomposing, algorithmic thinking, evaluating and generalising. In turn, a literature review by Kalelioglu, Gulbahar, and Kukul [27] showed that abstraction, algorithmic thinking, problem solving, pattern recognition, and design-based thinking were skills that were most frequently discussed within papers concerning definitions of CT. Furthermore, they found that the definition of CT itself often consisted of algorithmic and design-based thinking. Lastly, they concluded that several practices such as problem representation and decomposition could also be associated with CT.

Apart from what seems to be the core concepts associated with CT, Brennan and Resnick [10] also describe certain attitudes that support CT activities. As being described as 'CT perspectives' within their framework, these cover attitudes like expressing, connecting, and questioning. More specifically, a student should be aware of themselves, their connection with others and their technological surroundings.

To summarize, there are numerous definitions, complemented by frameworks, concepts, characterisations, skills and attitudes. The definitions as given by Selby and Woollard [41] and Kalelioglu, Gulbahar, and Kukul [27] come a long way in giving a broadly agreed on definition, and as such we will consider these as a starting point for this thesis. However, in order to come up with a more representative definition, we first need to explore the intersection between CT and mathematical thinking (MT), considering that the lesson series mainly revolves around mathematics education. As such, we will first explore the concept of MT, after which we will focus on the fronts CT and MT seem to intersect and how this enables CT integration in mathematics education.

2.2 Mathematical thinking

Similar to CT, the need to develop MT skills in students has seen increased attention in recent years. Already in 1963, Pólya remarked that it is needed that mathematics primarily learns students to think [37]. In the field, most researchers agree with this standpoint, and so Devlin argues that MT is *"a way of looking at things, by stripping them down to their numerical, structural, or logical essentials, and of analyzing the underlying patterns."* [15].

This is accompanied by the notion that MT is a way of looking at the world from a mathematical standpoint, and looking for logical explanations [43].

Although these definitions are generally agreed on, they are not specific enough in the sense that they provide concepts that can be associated with them. In an attempt to do so, Schoenfeld stresses that only mastering mathematical facts and procedures is not sufficient. He explains that thinking mathematically constitutes *"(a) developing a mathematical point of view - valuing the processes of mathematization and abstraction"*, and *"(b) developing competence with the tools of the trade, and using those tools in the service of the goal of [...] mathematical sense-making"* [40]. According to Schoenfeld, these 'tools of the trade' include abstraction, symbolic representation, and symbolic manipulation. These characteristics are also mentioned by other researchers, who associate multiple other characteristics with MT such as generalisation, problem-solving, evaluation, and reasoning about solutions and strategies. [9, 35].

If we closely look at the new Dutch math curricula, deployed since 2015, the definitions and concepts mentioned above are mostly apparent. They distinguish several MT activities such as: *"modelling and 'algebraizing', ordering and structuring, analytical thinking and problem solving, manipulation of formulas, abstracting, and logical reasoning and proving"* [14]. Drijvers tried to distil these activities into three core elements of MT: problem solving, modelling and abstraction [17], as can be seen in **Figure 2.1**. Here problem solving means being able to solve tasks for which a student

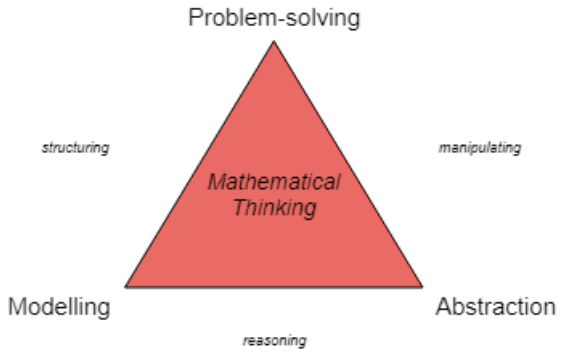

Figure 2.1: The three core elements of MT. Adapted [reprinted] from [17]

does not have a strategy yet [40]. Modelling implies the translation from a real-world problem into a mathematical problem, and vice versa. Lastly, abstraction is referred to as the isolation of specific attributes so that they can be evaluated independently from the other attributes [44].

An aspect of MT that is often overlooked, is algorithmic thinking (AT). This link with MT can best be described with the correspondence between AT and abstraction as described by Drijvers [17]. An important part of AT

is object formation, and within the abstraction process this plays a key role. In the abstraction process, reasoning on a high level about mathematical objects and their connections is essential. As such, simply following an algorithm is a procedure, whereas explaining and designing an algorithm requires object formation and generalisation.

For this thesis, we will use the definition as proposed by Drijvers [17], with an addition of the importance of AT and generalisation. In the next section, we will try to link CT and MT, and come up with a definitive definition of CT for this thesis.

2.3 Intersecting CT and MT

Before we try to link CT and MT using our findings in the previous sections, it is good to first touch upon the concept of contextualisation. This concept comprises the connection between real-world situations and the concepts that are associated with CT and MT. Kallia et al. [28] explain this concept with four categories of cognitive activities, which are depicted in **Figure 2.2**.

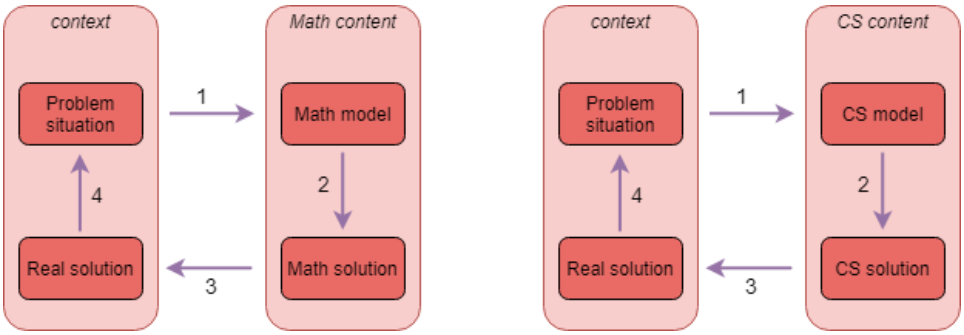

Figure 2.2: Contextualisation, concerning MT (left) and CT (right). Adapted [reprinted] from [28]

These activities entail (corresponding with the arrows): (1) the translation of a situation into either a mathematical or computational model, involving concepts like abstraction, modelling and pattern recognition; (2) the working and reasoning within mathematics and computer science; (3) the translation back to the original context, using the concept of generalisation; (4) the verification of the solution, verifying whether it solves the real-world problem adequately, involving the concept of evaluation. This concept already shows some similarities between CT and MT, but Kallia et al. take it one step further and consider the importance of the interplay between CT and MT within a mathematical context as can be seen in **Figure 2.3**.

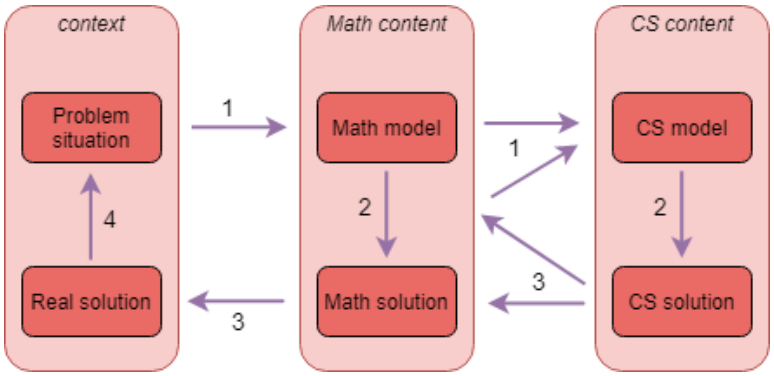

Figure 2.3: CT in mathematics. Adapted [reprinted] from [28]

This interplay between CT and MT has been previously researched, e.g. by Sneider, Stephenson, Shafer and Flick [42], and Weintrop et al. [49] who showed that problem solving, modelling, analyzing and interpreting data, statistics and probability were things that CT and MT have in common. Furthermore, Barcelos and Silveira [4] proposed, among other skills, that linking mathematical representations and algorithms is an important skill that can be developed when considering the interplay between CT and MT.

After a systematic literature review and a Delphi study, a consensus technique that seeks expert agreement on topics that have insufficient evidence [46], Kallia et al. proposed a definition of CT that relates to mathematics education. This definition is the final definition of CT that we will use for this thesis since it aligns with the findings in the previous sections. The definition that is given, accompanied by several thinking processes is as follows: CT is *"a structured problem-solving approach in which one is able to solve and/or transfer the solution of a mathematical problem to other people or a machine by employing thinking processes that include abstraction, decomposition, pattern recognition, algorithmic thinking, modelling, logical and analytical thinking, generalisation and evaluation of solutions and strategies"* [28]. This definition is visualised in **Figure 2.4**.

From the standpoint of our research, we want to highlight the concepts AT and generalisation. These concepts were most apparent within the course, which makes it worthwhile to deepen out these definitions. In addition, in **Table 2.3** we will briefly give definitions of the several other concepts as described in the definition by Kallia et al. which are apparent throughout the course.

2.3.1 Algorithmic thinking

Firstly we turn our attention to AT, which we already shortly touched upon within the previous section. Several researchers acknowledge that AT is not

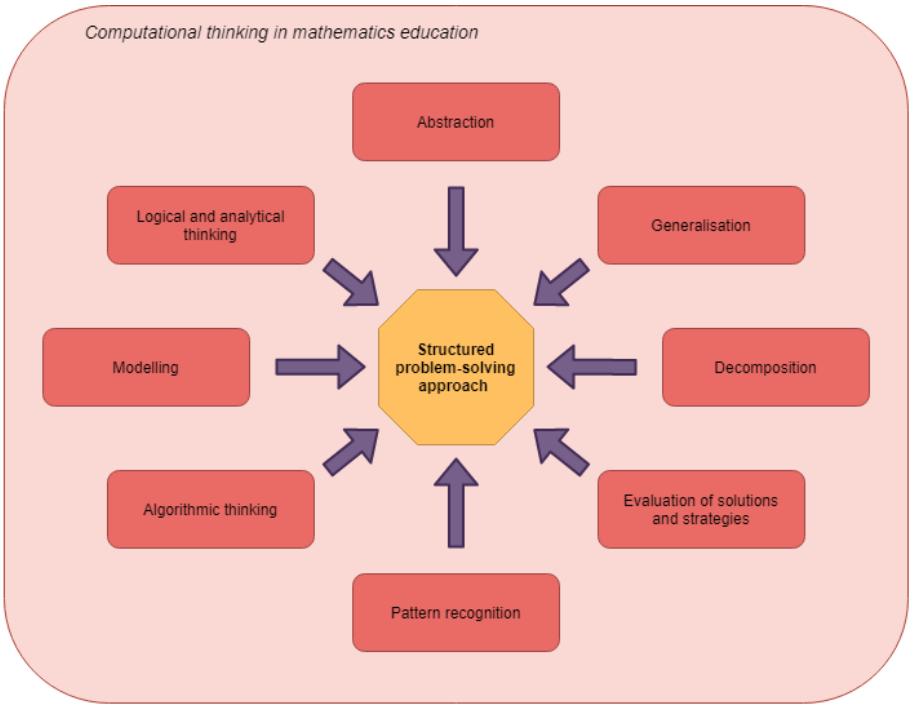

Figure 2.4: Visualisation of CT definition in mathematics education, as given by [28]

Table 2.1: Definitions of CT skills mentioned in [28]

Skill	**Definition**
Abstraction	Isolation of specific attributes so that they can be evaluated independently from the other attributes [44].
Decomposition	Breaking up a problem into smaller, more manageable parts [12].
Pattern recognition	Finding patterns and similarities across decomposed problems, to efficiently solve more complex problems.
Debugging	The evaluation and analysis of solutions through testing, verifying, reasoning and predicting outcomes [12].

only limited to the field of computer science [3, 25, 29]. They argue that algorithms arise from common activities that we do on a day-to-day basis, e.g. making a sandwich for your lunch. As such, the importance of learning skills that are central to AT is emphasised even more.

Kàtai defined AT as the skills that are required for processing or creating algorithms, where 'algorithms' refer to procedures for solving a problem step-by-step using several finite, implementable operations [29]. Cooper, Dann and Pausch associated several skills such as decomposition, usage of basic data structures and logical structures, repetition, and generalisation [11].

Instead of several concepts, Futschek came up with several abilities

related to AT to understand and construct algorithms. He defines that these abilities of AT include *"the ability to analyze given problems; specify a problem precisely; find basic actions that are adequate to that problem; construct a correct algorithm using these actions; think about all possible special and normal cases of the problem; and improve the efficiency of an algorithm"* [19]. Later, Futschek and Moschitz added that the use of commands and sequencing could help in adequately constructing algorithms. Examples of such commands are iteration (e.g. loops), and alternative (e.g. conditional statements) commands. Reasoning about algorithms and considering special cases is also part of AT [20]. For our research, we will closely follow this definition by Futschek. In our view, AT consists of the following abilities: (1) construct algorithms by coming up with step-by-step procedures, (2) read and execute an algorithm, and (3) reason on a higher level about an algorithm through evaluation which leads to insights for improving algorithms.

2.3.2 Generalisation

As we have seen, both computational and mathematical thinking consider the concept of generalisation. According to Mason, *"generality is central to all of mathematics"* [33]. However, he also notes that many educators and experts are unconsciously so used to using generalisation, that they have become oblivious to the importance of teaching this thinking process to novices. Dumitrascu acknowledges this problem and stresses the powerful nature of generalisation as a thinking process. She characterised generalisation as *"a statement that is true for a whole category of objects; it can be understood as the process through which we obtain a general statement; or it can be the way to transfer knowledge from one setting to a different one"* [18]. Selby and Woollard mostly agree with this definition, but they also highlight the importance of decomposition. According to them, generalisation is the recognition that smaller pieces of a problem (decomposition), can be used and applied to unique or similar problems [41].

2.4 Assessment of CT

As a consequence of the earlier mentioned lack of a formal definition of CT amongst scholars, there is also no consensus on how to properly assess CT. Differences in assessment of standalone CT activities and subject-integrated CT activities, further complicate the development of proper generally agreed on methods to assess CT. It is worthwhile to look at past research of subject-integrated CT activities and their assessment methods, to enable us to make an informed choice about which methods to use for our research.

Brennan and Resnick [10] linked CT with the programming in Scratch, an environment that provides students aged 8 to 16 with the tools to pro-

gram their own interactive stories, animations and games [34]. They identified that Scratch contains several CT concepts, which they in turn coupled with several CT practices and perspectives. To capture the presence of these, they used project portfolio analyses, artefact-based interviews, and design scenarios. Interesting was that the design scenarios, where students needed to fix a bug, shed more light on the process-in-action rather than the process-via-memory that was present in the artefact based interviews. Finally, they gave some suggestions to better assess CT in Scratch and other (programming) design activities. On the topic of artefact-based interviews, they noted that further learning should be supported, multiple artefacts should be examined to solidify assumptions, and the learner's process also should be considered. On a more general note they argued that checking in on multiple points in time and engaging teachers, peers and parents in the assessment could be beneficial. Seeing that CT is a process and considering there are several levels of knowing certain CT concepts and practices, in our view this argument is well-founded.

Hutchins et al. [26] created a learning environment that aided in teaching physics to high school students called the Collaborative Computational STEM learning environment (C2STEM). C2STEM uses a computational modelling approach, that combines visual model building and a domain-specific modelling language. In an effort to analyze the student learning before, after and during the use of C2STEM, they used several assessment methods. Firstly they used pre-post test assessment, which contained both physics and CT related questions. Furthermore, they employed Preparation for Future Learning (PFL) assessment which captured the ability of students to adapt to new situations, regarding problem-solving strategies that CT provides. Lastly, video analysis of screen recordings of student work and conversations was used, where they mostly looked at model-building approaches and difficulties they had during the tasks. Using these methods, they found that the students that used C2STEM achieved a better understanding of physics/CT concepts and practices than students who learned utilising the traditional curriculum. Particularly students that used a step-by-step model building approach seemed to develop this deeper understanding. Lastly, they noticed that students were able to transfer their newly learned concepts and problem-solving skills to solve new problems.

Waterman, Goldsmith and Pasquale [47] identified three stages of CT integration: (1) identifying existing connections (2) enhancing and strengthening connections (3) include activities that are targeted to develop CT skills. In light of this, they tried creating material that both targets the development of CT, and at the same time supports learning in the subject that it is integrated into. They designed an integrated module (I-Mod), building on the science module "Survival and Organisms". The main activity for this module that proved fruitful for integrating CT was *Oh Deer!*, which learned students about the survival of a population if the available resources do not

meet the needs. This activity could be subdivided into separate, concurrent stages: Modelling the ecosystem and generating simulation data; representing and analyzing data; and lastly using a complex model. The researchers mainly relied on observations during the activities, analysis of conversations between students, and student-teacher interactions. They concluded that students expressed ideas that demonstrated both CT and science understanding. Lastly, they acknowledged that unfamiliarity with technology could hamper the development of both CT and science-related content.

Weintrop, Morehouse and Subramaniam [50] looked at how in a library setting, fostering CT through programming could be assessed. Firstly, they found that because CT and fostering CT through programming in libraries is still quite new, the assessment of these things is difficult. Furthermore, from interviews that they held with library personnel, it became apparent that the most common methods to assess CT development were taking attendance and tracking retention across sessions, observations, surveys or questionnaires, impromptu interviews, and anecdotal feedback. Here they righteously noted that, although it was mentioned most, attendance and retention gives little insight into attitudinal or learning outcomes. Lastly, they also stated that a library setting could be better to learn CT, since this is more focused on enjoyment and engagement, whereas a school setting often results in a lack of motivation for students. This is also where they argued that assessment within schools needs to be either engaging or embedded within the activity itself.

From the research given above, we can see that assessment is often specifically made for the activity itself. In an effort for a more general approach, Román-González [38] came up with the CT-test (CTt). This consisted of 28 multiple choice questions that each were addressing some computational concept. He found that this test functions as a static and decontextualised assessment, which makes it a complementary assessment to e.g. C2STEM [39]. However, it must be noted that this test only covers the computational concepts as stated by Brennan and Resnick [10], and leaves out computational practices and perspectives. Similar to the CTt, two other noteworthy assessment methods exist: (1) the Test for Measuring Basic Programming Abilities (TMBPA) by Mühling, Ruf and Hubweiser [36], and (2) Commutative Assessment (CA test) by Weintrop and Wilensky [51]. These also assess the CT concepts as proposed by Brennan and Resnick.

In conclusion, it seems that there is no assessment method that is able to fully capture CT development. This is emphasised by Grover [23], who argues that in order to find a comprehensive understanding of CT skills development, there needs to be a so-called system of assessments that combines different types of assessment methods. From the literature that has been discussed in this section, we can see that common assessment methods include pre-post tests, evaluation of student work, interviews, and (analysis of) observations. In this thesis, we will try to closely follow these assessment

methods.

2.5 GeoGebra

In the previous section, we have seen some examples of subject-integrated CT activities. Although unplugged activities to foster CT has seen increased attention over the past years [7], most research involves some computer program or tool. This is not surprising since CT is very much related to computer science [53]. For our research, we resort to the digital tool GeoGebra. According to GeoGebra, their tool is *"dynamic mathematics software for all levels of education that brings together geometry, algebra, spreadsheets, graphing, statistics and calculus in one easy-to-use package"* [21]. As was emphasised in **Section 2.3**, AT plays an important role in both CT and MT. The created algorithm to solve a problem can easily be outsourced to a digital tool, such as GeoGebra. Furthermore, as will be made clear in the next chapter, the core of pure mathematics for the age group that the research revolves around is algebra, calculus and analysis of functions. Standard functions (e.g. polynomials) and standard procedures (e.g. finding the derivative) may already be things that students are familiar with, using pen and paper. However, outsourcing this work to GeoGebra in the form of designing efficient and flexible algorithms may prove to be fruitful. Moreover, since GeoGebra for a large part relies on generic functions, it will require students to reason on a higher level about object formation, which was pointed out by Drijvers to be a major part of MT and abstraction [17].

An important part of GeoGebra that goes unnoticed is the visualisation aspect. According to Adelabu, Makgato, and Ramaligela [2] a multimedia environment that also functions as a visualisation tool, encourages and enhances the exploration of mathematical concepts. This makes it possible for students to analyze data and perform better calculations, which results in a more permanent and effective learning process. Futschek also acknowledged this potential, with the addition that it can help students that struggle with learning AT. These students often struggle with abstraction, because computers work differently than humans think. He argued that through visualisation and manipulation, students are able to bridge the gap between the real world and a programming environment [20].

2.6 Research questions

From the literature that has been discussed in this chapter, we now have a better understanding of concepts central to CT and MT such as algorithmic thinking and generalisation, and methods for assessing CT. This leads to the definition of the research question that will be addressed within this thesis. We have defined the following main research question and respective

sub-questions that help in answering the main research question.

MRQ: *What characterises the use of algorithmic thinking and generalisation skills by 16-17 year old secondary school students after a 5-lesson calculus course that focused on the use of the mathematical tool GeoGebra?*

SRQ1: *In which data sources is algorithmic thinking and generalisation apparent?*

SRQ2: *What are the learning outcomes of students regarding algorithmic thinking and generalisation skills after a 5-lesson calculus course that focused on the use of the mathematical tool GeoGebra?*

SRQ3: *What challenges did students encounter during the learning process?*

Chapter 3

Methods

Within this chapter, we will look at the different methods that have been used within our research to assess CT development, in particular related to AT and generalisation. First, we will provide the intervention design, in which we will elaborate on objectives, learning materials and tools, and the activities of the lesson series. Next, we will elucidate the different data sources that have been used and how this data was collected. Lastly, we will give some insight into how the collected data has been analyzed and triangulated as a means to find patterns across multiple data sources.

3.1 Intervention design

For our research it is needed that we elaborate on the intervention design. This includes the objectives of the NRO regarding this lesson series, an educational context, a description of the participants and others involved in the lesson series, and insight into the circumstances and learning activities of the lesson series. Lastly, we will give an inventory of the materials and tools throughout the course.

3.1.1 Objectives of the lesson series

As part of a larger effort to tackle the problem of fostering both MT and CT in an increasingly technology-rich environment, the NRO is conducting a multi-phase design study that focuses on how the use of digital tools can aid pre-university students with the development of CT skills related to MT in a calculus course. At the time of writing, the study is in its first of two design cycles. For that matter, this thesis also functions as a feed-forward to the second design cycle. Their study aims to extend the knowledge on how to address learning goals concerning CT and MT within Dutch upper secondary education. Within the study, educational researchers and teachers from the field of computer science and mathematics tried to translate

notions that emerged from literature into practical guidelines. According to the NRO, this interplay between teacher and researcher within the design process ensures that the results of the study are easily accessible to teachers, and also guarantees that it is suitable for improving teaching practices. The results of the study that are relevant to our thesis consist of (1) learning activities that involve the use of digital tools and aim to teach key aspects of CT and MT, suitable for pre-university students that follow pure mathematics courses, and (2) assessment instruments that are able to assess learning outcomes related to CT and MT. Within this thesis, we will focus on the outcomes of the learning activities from the first design cycle. Furthermore, we will also turn our attention to the effectiveness of assessment methods used for the learning activities.

3.1.2 Educational context

For this research, we focused on one school of a teacher that is part of the research consortium. This school with 1400 students, is situated in the west of the Netherlands in a sub-urban town with approximately 24.000 inhabitants. For every class 5 60-minute lessons stretched over 2 weeks were designated for the lesson series, from the 12^{th} of March 2021 until the 26^{th} of March 2021. These took place during mandatory computer science lessons for the participants.

The main goal of the lesson series is to learn students to solve problems related to calculus within GeoGebra. This mostly consists of employing algorithmic thinking and generalisation skills such as constructing and evaluating algorithms within GeoGebra, usage of logical structures and generalising from specific to general solutions.

3.1.3 Learning activity

Before we dive into the activities that have been part of the lesson series, it must be noted that this research has been subject to special circumstances. Since the end of March 2020 the world, and in our case the Netherlands, has been experiencing a pandemic due to the COVID-19 virus. This meant that schools were closed down or heavily restricted in giving lessons in person, which meant that all lessons had to be held online and students had to work together online as well. This asked for slight alterations in the supply of the lessons and learning materials to the participants. With the use of Microsoft Teams, both the lessons and learning materials could be offered in an orderly fashion. Apart from this, the data collection also had to be altered slightly. This will be elaborated in their respective subsection(s).

Furthermore, we again want to stress that this lesson series was designed by teachers and educational researchers from the research consortium. We

did not have any involvement in the design of the materials or implementation of the activities.

At the start of the lesson series, students were asked where possible to form duos. Moreover, if students were unable to make a duo, the teacher would do this instead. During the lessons, students would work together with their partner(s) by making the exercises in the workbook and GeoGebra. It was the responsibility of the students to work their way through the paragraphs. The teacher had set guidelines for which paragraphs had to be finished before the next lesson, but students themselves had to take responsibility to have finished everything after the 5 lessons.

Lessons started in a general video meeting where the teacher would give some general info about the lesson. After this students would move to separate channels. These channels were made for each duo or trio in which students could share their screens and work together in peace.

The activity was led by 1 male instructor, that also was the teacher of the class. He has been teaching at this school since 2003 and is currently teaching both mathematics and computer science. His function mostly consisted of explaining the activities, checking in on students during the activities, and answering questions from the students that encountered problems during the activities.

3.1.4 Materials and tools

Firstly, a student workbook that was designed by the research consortium that consisted of 7 paragraphs was used for the lesson series. A link to this workbook can be found in **Appendix A**. In combination with GeoGebra, this workbook functions as the core of the lesson series. In **Table 3.1** the content of each paragraph is briefly explained.

The general idea is that the content of the lesson series is provided in a way that also encourages the use of AT and generalisation skills. For example, students are required to make the exercises and come up with algorithms on paper first, after which they implement it in GeoGebra using logical structures. These structures consist of sequences of steps, conditional statements and iteration based commands. Furthermore, in every paragraph there is a notion of generalisation by guiding students from a specific to a general solution. As such, in every paragraph students are encouraged to use their AT and generalisation skills to solve the given exercises.

As mentioned, GeoGebra was used as a complementary tool to the workbook. In **Section 2.5** we already provided some information on GeoGebra. Participants had access to this tool either via a desktop app, or a web-based version of the program. Both of these versions had the same features and were only differing in aesthetics.

Lastly, both a pre- and post-test were used to assess several computational thinking concepts. These tests were also designed by the research

Table 3.1: Student workbook paragraph overview

Paragraph	Title	Description
1	Line through two given points	Students come up with an algorithm to draw line through two given points in GeoGebra, generalize this algorithm to all possible points and handle special cases with the use of conditional statements
2	Perpendicular bisector	Students come up with an algorithm to have the perpendicular bisector of two given points drawn in GeoGebra, generalize this algorithm to all possible points, and handle special cases with the use of conditional statements.
3	Centre of gravity	Students come up with an algorithms to determine the centre of points, medians and centres of gravity in GeoGebra. This algorithm is then generalized for all possible triangles and special cases are handled with the use of conditional statements.
4	Tangent to a parabola	Students come up with an algorithm to draw the tangent to a standard parabola in GeoGebra, which is then generalised for all possible parabolas.
5	Bundles of tangents to parabola	Students come up with an algorithm to draw a bundle of tangents to a standard parabola using iteration commands. This algorithm then is generalised for all possible parabolas.
6	Tangents to other graphs	Students alter the algorithm from paragraph 5 for graphs of functions other than quadratic functions.
7	Zero points with tangents: Newton-Raphson	Students come up with an algorithm for approximating zero points according to the Newton-Raphson method, with the use of iteration commands and macros.

consortium and a link to these can be found in **Appendix A**. The pre-test consisted of 6 questions, whereas the post-test consisted of 6 questions similar to the pre-test and 4 questions about applying acquired GeoGebra knowledge from the lesson series to new situations. In **Table 3.2**, the CT concepts that are being assessed for every question is shown. The concepts that are mentioned, have been explained in **Section 2.3**.

3.1.5 Participants

Divided over two 11^{th} grade classes, in total 20 pre-university students aged 16-17 were designated as participants. Of these 20 students, 4 were female and 16 were male. These two classes were similar to each other since they shared the same background in programming and mathematics, as well as having the same teacher. All the students chose the course computer science, in which they previously programmed with JavaScript. Furthermore, they also all followed the course math B, the equivalent to pure mathematics. For this reason, we will consider these two classes as one group. All students were provided with the same tools and teaching materials.

Table 3.2: CT concepts per question

Question	CT concepts	Question	CT concepts
Pre/Post 1	Problem solving Decomposition	**Post 7**	Problem solving
Pre/Post 2	Algorithmic thinking Repetition structure	**Post 8**	Algorithmic thinking Repetition structure Abstraction
Pre/Post 3	Pattern recognition	**Post 9**	Problem solving Algorithmic thinking Abstraction
Pre/Post 4	Algorithmic thinking Repetition structure Evaluation	**Post 10**	Problem solving Algorithmic thinking Abstraction
Pre/Post 5	Algorithmic thinking Pattern recognition		
Pre/Post 6	Algorithmic thinking Pattern recognition Repetition structure		

3.2 Data collection

For our research, we relied on qualitative data from four data sources. Similar to an experiment with the same content by Borkulo et al. [45], these data sources included filled in student workbooks, completed GeoGebra files, and student mini-interviews. Additionally, we also made use of student answers to the pre- and post-tests. It must be noted that since we are part of a larger research effort, the data collection not only has been handled by us. For example, due to time constraints multiple interviewers were involved in conducting the interviews. Furthermore, due to the circumstances noted in **Section 3.1.3**, the data collection has entirely taken place online. The research consortium has made available an online data storage, which was used to store all data that was collected during the research. Before the lesson series started, every student was asked for consent for the collection of their digital work and consent for the recording of audio and/or video during the interviews. We will now look more closely at each of the data sources.

3.2.1 Pre- and post-tests

The pre- and post-test as elaborated in **Section 3.1.4** were held at respectively the beginning and the end of the lesson series. For both tests, students were asked to hand in their answers either in a text document or by making a scan of the answers they had written down on paper. Students were given

30 minutes to complete the pre-test and 40 minutes for the post-test. If students did not manage to complete the test in the designated time, they were asked to complete and hand it in outside of the lesson. Students were able to hand in the tests through their own directory within Microsoft Teams that the teacher had made available to them. We and other researchers then uploaded all of the pre- and post-tests to the online storage of the research consortium.

Seeing our thesis focuses on AT and generalisation, we will focus on these concepts of the pre- and post-test. It became apparent from **Table 3.2** that pre- and post-test questions 2, 4, 5, and 6, and post-test questions 8, 9 and 10 address these concepts. These are the questions that we will focus on during our analysis of these tests.

3.2.2 Student workbooks

As was mentioned in **Section 3.1.4**, the workbook was provided online. Students were given a digital version of the workbook in their own directory in Microsoft Teams. They were asked to either fill in their answers in this digital version or provide photos/scans of their answers that they had written down on paper. In case of scans or photos, we asked students to keep them within one file for the sake of clarity.

These workbooks were used to identify how students solved problems related to calculus, while also appealing to AT and generalisation skills. At the end of the lesson series, students were asked to hand in their final version of the workbook. We then collected every workbook from the respective directories and uploaded them all to the online database of the research consortium.

3.2.3 GeoGebra files

Exercises in the student workbook required actions in GeoGebra. This resulted in one file where all actions from that paragraph were combined. Thus theoretically, every student should have 7 GeoGebra files at the end of the lesson series. Students were able to easily export these files within both the desktop app and web-based version. Students were asked to upload all their files to their designated directory in Microsoft Teams. From there, we were able to download all files and upload them to the online database of the research consortium.

In combination with the workbook, these files will give a better understanding of the process and encountered problems of the students in solving the tasks in the workbook that require the use of AT and generalisation skills.

3.2.4 Student interviews

In lesson 3 and lesson 4 mini-interviews with the participants were held. Since these interviews had to be done during the lessons, we encountered some time limitations. In order to still capture all experiences from students, two researchers from the research consortium helped in conducting the interviews (hereafter called Researcher A and Researcher B). The allocation of students to the author of the thesis, Researcher A and Researcher B can be found in **Table 3.3**. Every duo or trio of students that gave consent,

Table 3.3: Allocation of students for conduction of interviews

Researcher	**Allocated students**
Author of thesis	Student3 and Student4, Student9, Student10 and Student 11, Student14 and Student15
Researcher A	Student5 and Student6, Student12 and Student13, Student16 and Student17
Researcher B	Student1 and Student2, Student7 and Student8, Student18, Student19 and Student20

was interviewed in a 10 to 15-minute session. In total, 19 students were interviewed. We followed a semi-structured interview guideline that was specifically made for this lesson series by the research consortium. These questions can be found in **Figure 3.1**. Questions were divided into general questions, questions specifically for tasks halfway, and questions after completion of all paragraphs. These guidelines offered some structure during the interviews, but it was also needed to respond to the answers students gave. By doing this, we hoped to gain more fruitful comments.

The interviews were held in the Microsoft Teams channel of the duo or trio concerned. Before the interviews started, we gave the students some assurance that they should not be afraid that their answers would be wrong or weird. We stressed this since students might be reluctant to speak their minds in the presence of both us and their peers. During the interview, we also tried to involve all students to get as many different answers to our questions.

At the start of the interview, we started a recording using the capture software Open Broadcaster Software (OBS). This software records the screen, the audio played through speakers/headphones and audio that is picked up by the microphone. After all groups were interviewed, we edited all recordings together and uploaded this to the online database.

These interviews were intended to give us some insight into student opin-

General:

1. How did it go? What did you run into (regarding GGB, formulas, syntax, conditions, etc)? How were you able to solve the problems that occured (workbook, alone, peers or teacher)?
2. What do you think about GGB? What advantages/disadvantages do you see?
3. How was it to first write down your steps and then work in GeoGebra?

On the assignments

After some task halfway:

1. Did you reuse previous work? (show GeoGebra)
2. How was it to define variables and functions in GeoGebra and using commands? (show GeoGebra, if-then-else, row, macro, iterationlist)
3. How did you make the step from a specific to a general solution? (show GeoGebra)
4. How did you evaluate you were done? (show GeoGebra, exception(s))
5. Would you do anything differently next time?

At the end:

1. Look back at Chapter 1-3. How did it go? What did you encounter? (show GeoGebra, use questions a – e)
2. Look back at 4-6. How did it go? What did you encounter? (show GeoGebra, use questions a – e)

Look back at Chapter 7:

3. How did you fill in the flow chart to describe the NR method? (7.1, show workbook/paper)
4. How do you think you can use GeoGebra to approximate the other zero point using the Newton-Raphson method? (7.9)
5. How did you go about approximating the golden ratio? How did you reuse your earlier solution? (7.10)
6. How did exploring the effect of changing the starting value go? When does it go wrong? (7.11)

General at the end:

1. Can you describe what you learned in general in this lesson series and how it might help you in solving problems in the future?
2. Can you describe computational thinking?

Figure 3.1: Semi-structured interview guidelines regarding the lesson series

ions about the lesson series and GeoGebra, strategies in solving the exercises in the workbook and GeoGebra, and student attitudes towards several CT concepts and practices.

3.3 Data analysis

In this section, we will explain the different methods that were used for the analysis of the collected data. We will also elaborate on the triangulation of the data, and how this aids us in answering the research questions of this thesis.

3.3.1 Pre- and post-tests

For the analysis of the pre- and post-tests, we first made sure that all students that were part of the same duo or trio were grouped. This made sure that in a later stage, we could easily distinguish students that worked together.

After this, we started with assessing all the answers from the pre- and post-test that a student gave. We used a 5-point assessment scale, that was designed together with the tests themselves by the research consortium. This

5-point scale allowed us to better handle different answers that students may give. As an example of the assessment scale, in **Figure 3.2** the assessment for question 4 of both the pre- and post-test is depicted. For the assessment scales of the other questions, we refer to the link of the pre- and post-tests in **Appendix A**. In order to give an orderly overview of all assessments, we

Assessment	
Excellent	Student gives correct answer A (pre), A and D (post). Student explains that color (variant pre) or size (variant post) of the circle is chosen once. Possible explanation that B is also **possible** with this packing machine.
Good	Student gives correct answer A (pre), A and D (post). Student explains that colour (variant pre) or size (variant post) of the circle is chosen once. And B is mentioned next to the correct answers as being **not possible** and that choice is explained.
Adequate	Student gives correct answer A (pre), A and D (post), possibly including B with no or unclear explanation OR student only mentions that B is **not possible**, with clear explanation.
Marginal	Student gives wrong answer with (good) explanation. Or only that B is **not possible** without explanation.
Inadequate	Student gives wrong answer and no (good) explanation

Figure 3.2: Assessment scale of pre- and post-test question 4

chose to link the numbers 1-5 to Inadequate - ... - Excellent respectively. If a student did not provide an answer or some entry was simply missing, we denoted this with a − sign. By also using colouring, we thought that it would help to achieve a clear visual overview of the results. In the end, we will have a table containing all scores for a specific student. In addition, we also dedicated a comment section to the analysis, which allowed us to add comments about remarkable or interesting answers from students.

3.3.2 Student workbooks

We started by grouping the students that worked together, for a clear overview. In order to analyse the student workbooks, we first had to come up with an appropriate assessment scale. In contrary to the pre- and post-tests, here we opted for a 3-point scale. For every paragraph, we wanted to reflect if students understood and correctly executed the tasks within the workbook. We opted to assess the paragraph as a whole since tasks within a paragraph often covered the same content. This scale can be found in **Figure 3.3**. Similarly to the pre- and post-test, we assigned scores of 1-3 to Inadequate - ... - Good respectively. If students did not provide an entry for the paragraph, this would be denoted by an − sign. This assessment process was done for every student and concerned every paragraph. This resulted in a table where every student would be awarded a score for each paragraph

Assessment	
Good	Student understands the core concepts of the paragraph, and provides answers that adequately reflect the tasks
Adequate	Student understands the core concepts of the paragraph, but makes a few mistakes.
Inadequate	The student has provided incomplete answers to paragraph in workbook **OR** does not seem to understand the core concepts of the paragraph

Figure 3.3: Assessment scale for paragraphs of the student workbook

in the workbook. Additionally, we also provided a brief justification of the score that was given. These justifications would in a later stage of triangulation give more insight into what might go right or wrong for students in for example the translation between the workbook and GeoGebra. Lastly, we will provide a short overview of encountered issues or mistakes which have become apparent from the workbooks.

3.3.3 GeoGebra files

Similarly to the analysis of the student workbooks, for the GeoGebra files we again opted to use a 3-point assessment scale. This scale can be found in **Figure 3.4**. After grouping all the students that worked together, we

Assessment	
Good	Student understands the core concepts of the paragraph, provides a GGB file that adequately reflects the workbook tasks
Adequate	Student understands the core concepts of the paragraph, but makes a few mistakes.
Inadequate	Student has provided incomplete GGB file **OR** does not seem to understand the core concepts of the paragraph

Figure 3.4: Assessment scale for GeoGebra files per paragraph

started analysing every GeoGebra file that was associated with a certain paragraph using the assessment scale. We looked at the tasks that were given in the student workbook that concerned actions in GeoGebra and analysed if these tasks were correctly executed in GeoGebra. This analysis resulted in a table where every student had scores associated with their GeoGebra files from a particular paragraph. In case students uploaded multiple GeoGebra files for one paragraph, these would be analysed as a whole. Again, we also

provided a brief justification of the given scores which would help is during triangulation. Lastly, we also included issues with or common mistakes in GeoGebra that became apparent from the analysis.

3.3.4 Student interviews

The first step in the analysis of the student interviews was the transcribing of the interviews. We transcribed our own portion of the interviews, and a student assistant that was part of the research consortium helped with the transcribing of the interviews that Researcher A and Researcher B conducted. After transcribing the interviews, we were able to start coding within the statistics program ATLAS.ti. This program focuses on the analysis of qualitative data.

Firstly, we used a codebook that was designed by the research consortium. This codebook consisted of multiple codes and subcodes. In **Table 3.3.4** a snippet of the codebook is given, where the code 'Problem solving' and its respective subcodes are shown. For the full codebook, we refer to **Appendix B**. With the use of these codes and subcodes, student answer-

Table 3.4: Snippet of interview analysis codebook, concerning the code 'Problem solving' and respective subcodes

Code group	Code	Subcode	Description
CMT aspects	Problem solving	Decomposition	This code is used when a student refers to breaking up a problem, using smaller parts or steps for better understanding. When only 'steps' are referred to, this is coded by 'Algorithmic thinking – steps'.
		Generalisation	This code is used when a student refers to a general solution, including 'first specific then general'.
		Evaluation	This code is used when a student expresses thinking of what is a correct answer, or how GeoGebra can be used to check the solution, see that it works.
		Strategies	This code is used when a student describes approaches of having an overview, reusing a solution, trying out in GeoGebra, using CT and a step by step approach to solve problems in general, keeping calm.

s/comments from the transcribed interview could be coded. This process started by taking one answer or comment at a time. We then proceeded by looking at both the context and content of the answer/comment. If applicable, we would add a (sub)code from the codebook to this answer. An example of this could be that a student would talk about checking if their answer in GeoGebra is correct. In this case, we could apply both the subcodes 'GeoGebra' and 'Evaluation'. In the remainder of this thesis, we will refer to these coded answers as quotes. Additionally, for every quote an identification subcode 'studentX' would be added. This was done to aid in

the triangulation process.

After all interviews would have been reviewed and coded, we would have an occurrence table of all (sub)codes. Accompanied with this, we would also have a full list of student quotes concerning these (sub)codes.

3.3.5 Triangulation

In previous sections, we often referred to analyzing our data sources in a way that would contribute to clarity in the triangulation process. This mostly consisted of clearly labeling and grouping students. The main goal of the triangulation process was to reveal, support or dispute assumptions that emerged from individual analysis of data sources. This ultimately results in coming up with conclusions that may contribute to the answering of the research questions.

We thus tried to find patterns or inconsistencies between the data sources. Using this method, we will try to gain a clearer insight into the learning outcomes and process of students. An example of an inconsistency could be that a student comments during an interview that they encountered no difficulties with the use of GeoGebra. However, after analysis of their GeoGebra files it may become apparent that they did in fact experience some difficulties. In order to find these patterns and inconsistencies, we need to essentially analyse all data sources of a student at once.

Chapter 4

Results

Within this chapter, we will present our results that emerged from our analysis of the pre- and post-tests, student workbooks, student GeoGebra files, and mini-interviews with students.

4.1 Pre- and post-tests

For the pre- and post-test we made a distinction between having handed in (1) both pre- and post-test, (2) only pre-test, and (3) neither pre- nor post-test. In **Figure 4.1** these hand-in rates are displayed. Since these

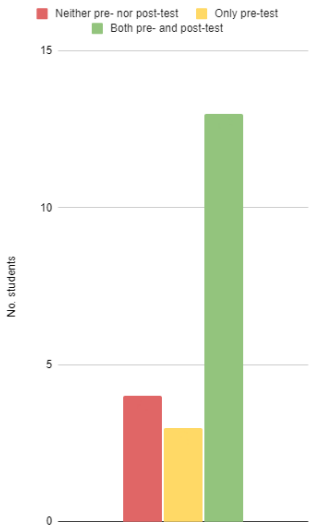

Figure 4.1: Hand-in rates pre- and post-test

rates resulted in only 65% of students handing in both pre- and post-test, we consider these rates on the edge of satisfactory.

As mentioned in the previous chapter, we will focus on the questions that concern AT and generalisation since these concepts of CT are most prevalent within the lesson series. Since the pre- and post-test questions are very similar to each other (e.g. pre-test question 2 corresponds with post-test question 2), we will look at both of these questions in tandem.

First, we will consider the general part of the pre- and post-test. This concerns questions 2, 4, 5 and 6. Noteworthy is that questions 2 and 6 require students to come up with their own algorithm, whereas questions 4 and 5 require students to follow a certain algorithm. The results from the analysis of these questions are given in **Figure 4.2**. Overall, scores for these questions are rather satisfactory. For every question, we can observe that scores remained rather stable, or slightly increased between pre- and post-test.

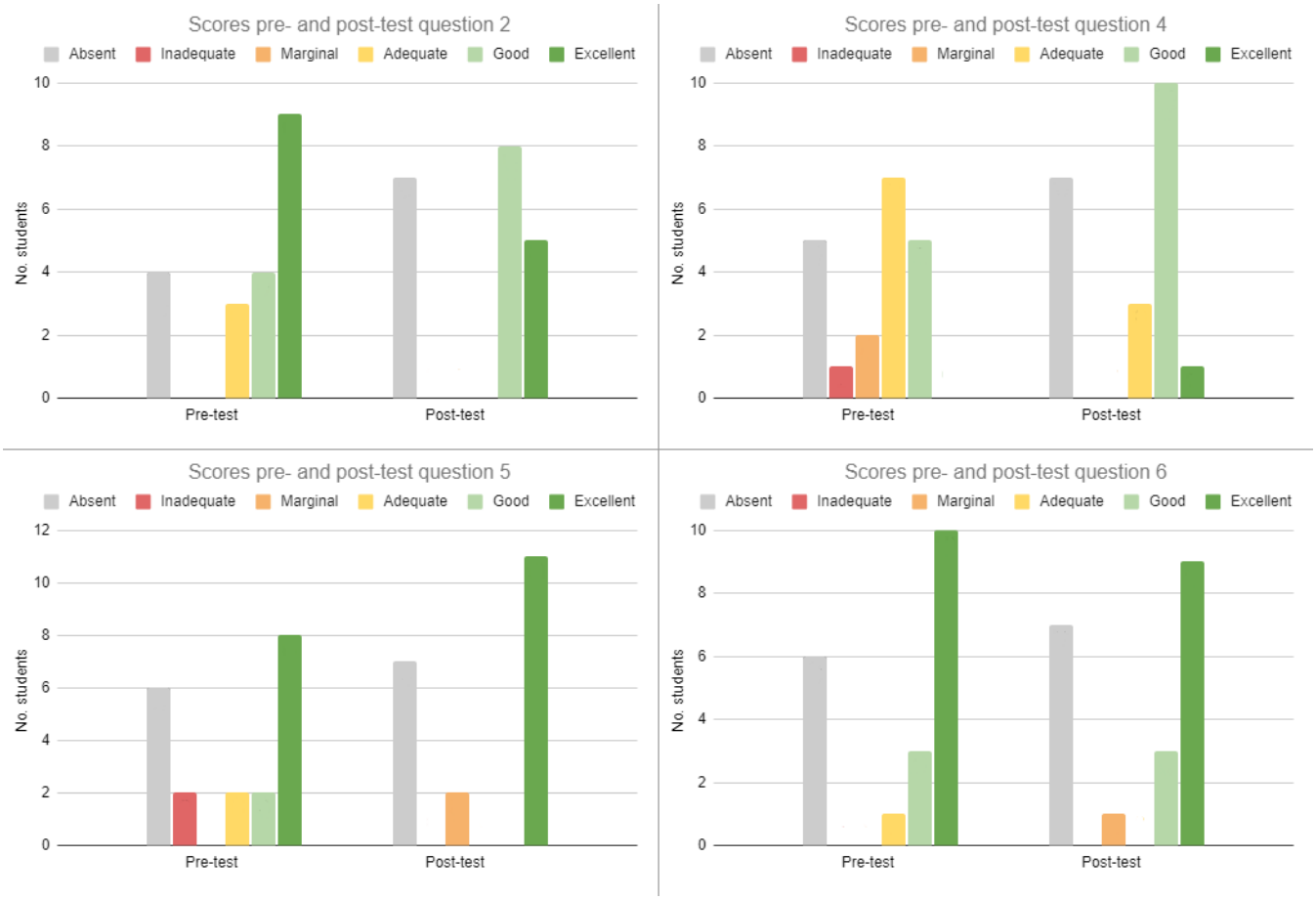

Figure 4.2: Scores for pre- and post-test questions 2, 4, 5 and 6

For question 2, students were asked to come up with a step-by-step guide for making a certain amount of pancakes (pre-test) or sandwiches

(post-test). For the pre-test 11 students mentioned repetition of some part of the steps, and 8 students discussed some conditional statement. For the post-test, again 11 students mentioned repetition, and 4 students used some conditional statement.

In particular question 4 seems to stand out, in the sense that it shows the largest difference in pre- and post-test scores. In this question, students were required to analyse an algorithm and evaluate which answers could have been produced by that algorithm.

Question 5, which was made significantly better as opposed to question 4, did not require students to give an answer based on evaluation of different answer possibilities. Here students needed to recognize the pattern which directed the students to the answer. For the pre-test 9 students recognized the pattern, whereas 11 students did in the post-test

In question 6, students were asked to come up with instructions that would lead a robot through a maze. This maze followed a certain pattern, that could be identified and repeated. Interestingly with this question, 6 students in the pre-test and 4 students in the post-test made use of a while-statement. This statement meant that some set of actions should be repeated until the maze exit. Furthermore, 2 students in the pre-test and 6 students in the post-test mentioned repetition of a set of actions. Lastly, 1 student made use of a flowchart in both the pre- and pos-test, which can be found in **Figure 4.3**

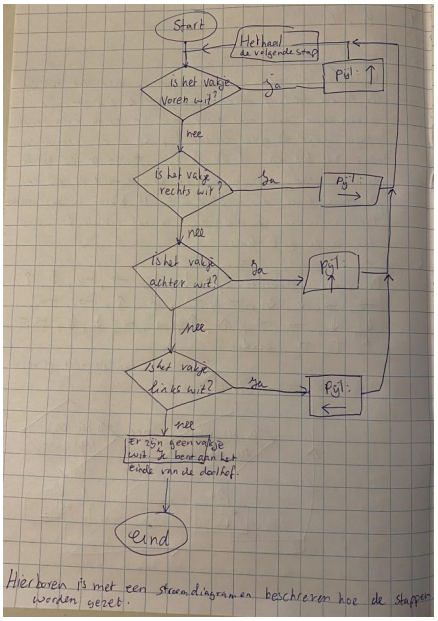

Figure 4.3: Flowchart from Student17 for pre-test question 6

In the specific part of the post-test students had to apply their acquired

GeoGebra knowledge from the lesson series to new problems. The results of our assessment can be found in **Figure 4.4**. We noticed that students struggled significantly more in comparison to the general part of the post-test. Questions 8 and 9 required students to come up with algorithms using commands from the exercises in the workbook and some new commands that were given in the question. Question 10 focused more on giving a clear step-by-step guide or flowchart, rather than the use of commands.

For question 8, we observed that 10 students were able to draw up a general plan of action. However, this often resulted in misuse of commands or wrong syntax. No student was able to present a correct solution for this question.

In question 9, we again noticed that 11 students were able to give a general plan but again did not manage to come up with a concrete algorithm. For this question, 1 student managed to come up with a sensible algorithm only making a little mistake with syntax.

Lastly, question 10 constitutes the best scores considering the specific part of the post-test. This question was rather similar to question 2, in the sense that students also needed to come up with their own step-by-step instructions. We observed that all 10 students that answered this question, managed to come up with a coherent set of instructions.

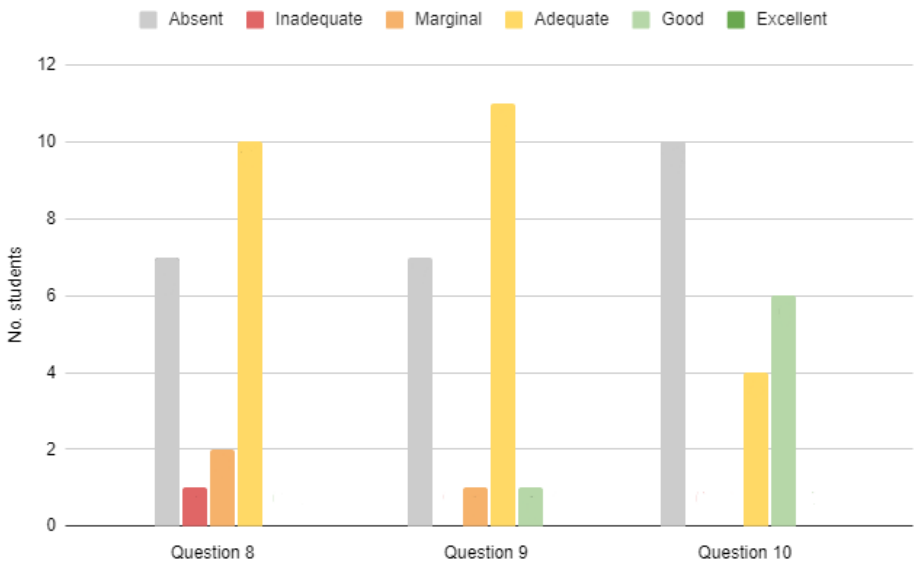

Figure 4.4: Scores for post-test questions 8, 9 and 10

4.2 Student workbooks

First, we noted down how many paragraphs a certain student had handed in. We distinguished between (1) none, (2) 1 to 3 paragraphs, (3) 4 to 6 paragraphs, and (4) all paragraphs. We have decided not to distinguish between correct and incorrect entries. These hand-in rates can be found in **Figure 4.5**. We observed that the individual hand-in rates were satisfactory since 65% (13 students) handed in 4 or more paragraphs and this coincides with findings in a similar experiment where on average 67% of students (10 out of 15) handed in their work [45]. To account for the 7 students that handed in 3 or fewer paragraphs, we also looked at the group hand-in rates which are also displayed in **Figure 4.5**. We made this choice since students often work together, resulting in only 1 student handing in their work. From these group hand-in rates we can see that from the 9 groups, 8 managed to hand in a complete workbook. In our eyes, this rate is even more satisfactory than the individual rates, since this might suggest that almost all students were involved in making the exercises in the workbook.

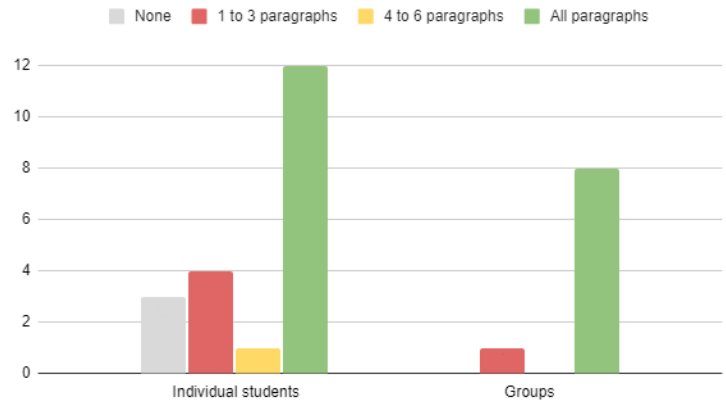

Figure 4.5: Hand-in rates student workbook paragraphs for individual students (left) and groups (right)

Next, we have the results of our assessment of all paragraphs that were handed in. These can be found in **Figure 4.6**. An interesting finding here is that students seem to have struggled significantly more with the tasks in the first two paragraphs, considering that for both of these paragraphs 6 students scored an 'Inadequate'. In paragraphs 3 and 4 we saw an increase in 'Good' scores and a decrease in 'Inadequate' scores. In paragraphs 5 and 6 we did not encounter any 'Inadequate' scores anymore but also noticed a slight decrease in 'Good' scores. Paragraph 7 saw a decrease in scores, where 3 students scored an 'Inadequate'. Looking more closely at these lower scores, we observed that students had a general idea of what to do, but were unable to come up with concrete, correct answers.

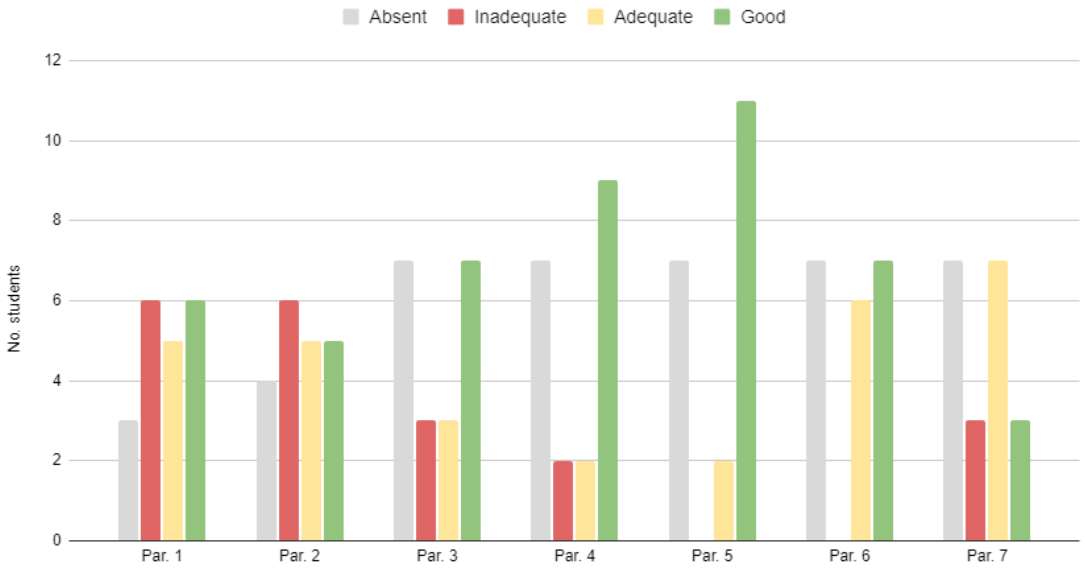

Figure 4.6: Student scores for the paragraphs of the workbook

4.2.1 Encountered issues

In our analysis, an important part was to identify common mistakes that students made during the exercises in the workbook. These are displayed in **Figure 4.7**.

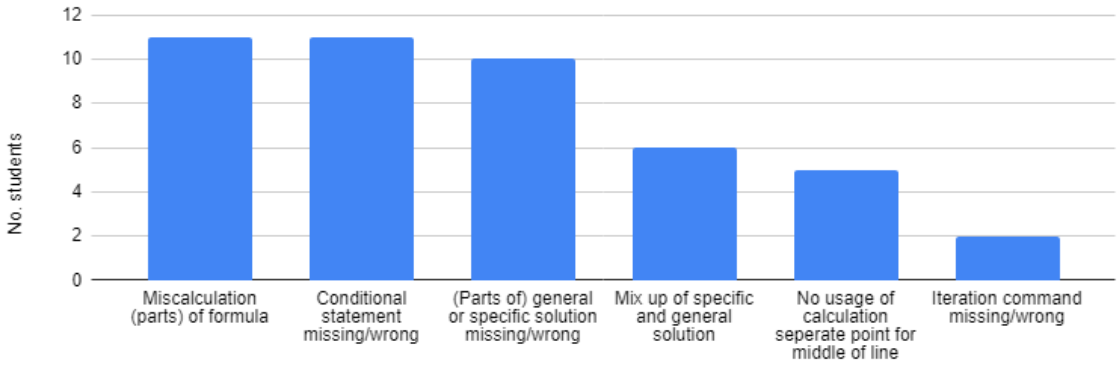

Figure 4.7: Observed mistakes from students based on the workbooks

Considering mistakes related to AT and generalisation, 11 students struggled with defining conditional statements. This coincides with our findings from **Figure 4.6**. Conditional statements were an important part of the first 3 paragraphs, and we observed that especially in these early paragraphs

students did not perform well. Furthermore, 10 students wrongfully defined parts of their specific or general solution. Additionally, 6 students seemed to not be able to convert from a specific to a general solution, and instead proceeded to use their specific solution. Lastly, 2 students were unable to come up with a correct iteration command.

Through deeper analysis, we observed that the majority of the above-mentioned mistakes arose from problems that are decoupled from AT and generalisation skills. We observed that by wrongly defining (parts of) their formulas (e.g. a line), students got confused when defining conditionals statements, iteration commands, and general solutions (11 students).

4.3 GeoGebra files

Similar to our analysis of the student workbooks, we commenced with an inventory of the hand-in rates for the GeoGebra files. We considered handing in a paragraph as providing all the necessary files containing the exercises of said paragraph. We again distinguished between (1) none, (2) 1 to 3 paragraphs, (3) 4 to 6 paragraphs, and (4) all paragraphs. These individual hand-in rates can be found in **Figure 4.8**.

The individual rates are not satisfactory, since on average 50% (10 students) did not manage to hand in a file per paragraph and we expected to see similar rates to those of the workbook. Furthermore, our average is below that of a similar experiment (on average 67% of students) which adds to this statement [45] Again, this may be subject to students working together with their partner(s) in one file. As such, the group hand-in rates are also displayed in **Figure 4.8**. Here we observed that 5 out of 9 groups managed to hand in all paragraphs, which is not much better than the individual rates. We thus stand by our previous statement that the hand-in rates for these files is not satisfactory. From the student interviews, we might find explanations for these low hand-in rates.

Next, we have the results of our assessment of all GeoGebra files associated with a paragraph. These can be found in **Figure 4.6**. From these results, we can see that the first paragraph did not pose many problems for the participants (10 students with score 'Good'). However, in paragraphs 2 and 3 we noticed a slight decrease in scores. Paragraph 4 saw a recovery in scores, after which in the remaining paragraphs scores were rather stable. After paragraph 4 no student scored an 'Inadequate' anymore, indicating that no big mistakes were made.

4.3.1 Encountered issues

In order to partly explain the scores that were given at the beginning of this section, it is again necessary to elucidate common mistakes that students made during the exercises in GeoGebra. Mistakes that were made regarding

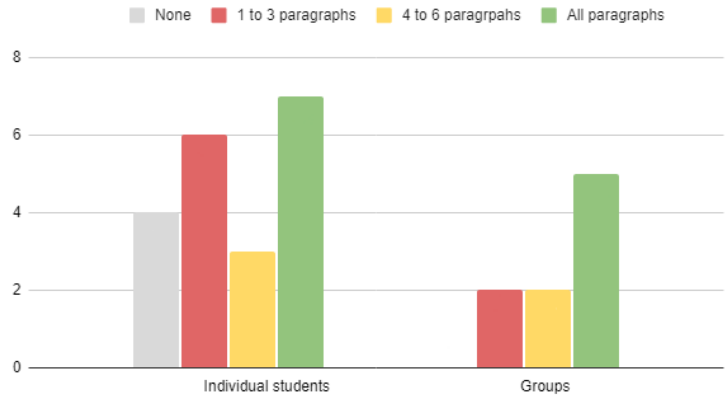

Figure 4.8: Hand-in rates GeoGebra files per paragraph for individual students (left) and groups (right)

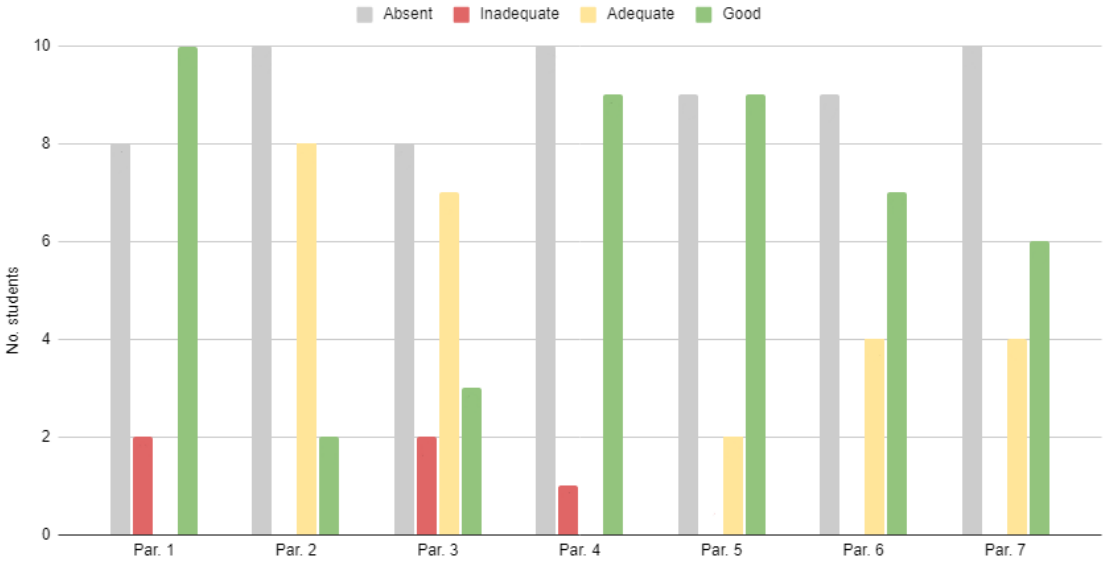

Figure 4.9: Student scores GeoGebra files per paragraph

AT and generalisation skills, consisted of lack of object formation that could help within exercises (10 students), stating unnecessary parts in their files (4 students), a wrong or missing iteration command (4 students), a wrong or missing macro (3 students), incorrect use of syntax (3 students), and a wrong or missing conditional statement (2 students). Furthermore, we also identified one other mistake that is less related to AT or generalisation. This mistake consisted of miscalculating (parts of) formulas (3 students), as we have also seen within **Section 4.2.1**.

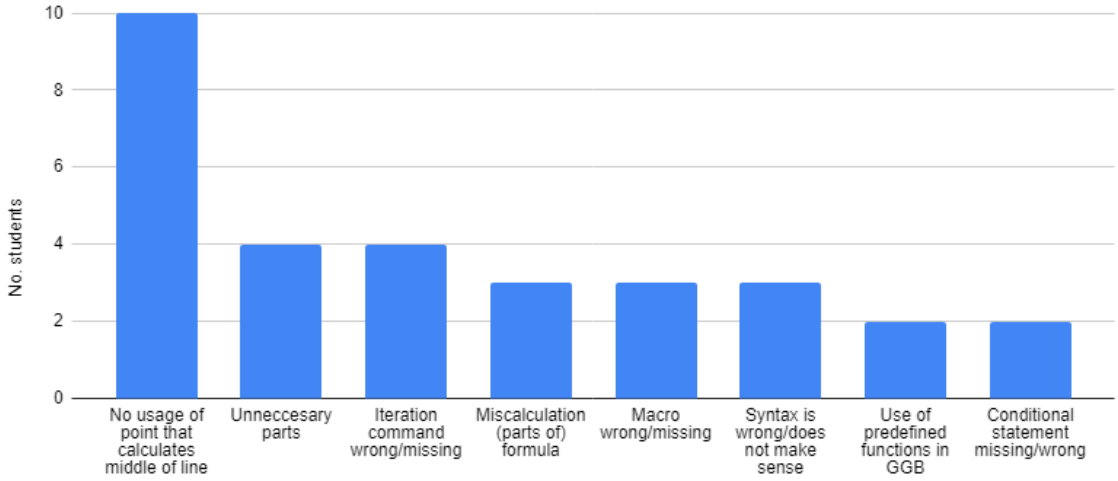

Figure 4.10: Observed mistakes from students based on their GGB files

4.4 Student interviews

For the interviews, we managed to interview 19 out of 20 students. It was preferable to have interviewed all of these students twice, once in lesson 3 and a second time in lesson 4. In **Figure 4.11** the rates of how many times students were interviewed are displayed. We managed to interview half of the students twice (11 students). Considering that in a similar experiment 80% of the students was interviewed, we are satisfied with having interviewed 95% of the students.

However, we did notice that it was difficult to engage all students within the conversation. Additionally, we observed that students were holding back with tapping into answers of other students and giving extensive answers. This resulted in less variable and detailed answers.

Furthermore, during the interviews in lesson 4, it emerged that a majority of the groups (6 out of 9 groups) did not make it past chapter 5. This meant that we were limited in asking questions about iteration and the use of macros, which were important parts of paragraphs 5 to 7.

After the interviews had taken place, we transcribed the interviews with help of other researchers. Next, we started coding these transcriptions using the codebook in **Appendix B**. In the following subsections, we will elucidate on what can be observed from these codes and accompanied student quotes.

4.4.1 Strategies regarding algorithmic thinking

We will first look at the subcodes of the code 'Algorithmic thinking'. We noticed that the subcode with the most quotations was 'steps', in which students express thoughts about algorithms or step-by-step procedures. This

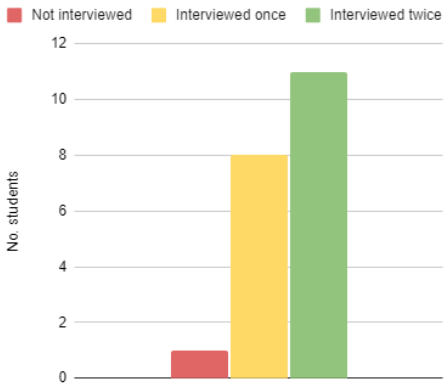

Figure 4.11: Rates showing how many times students were interviewed

manifested itself mainly through talking about the translation of step-by-step procedures that were necessary within the workbook, to the implementation of equivalent computational steps in GeoGebra. We observed that 7 students acknowledged the benefit of using step-by-step procedures.

> "Normally when I am stuck and I go through the steps calmly, I do get out, so yes the steps do help." (Student11, lesson 4)

> "It does make you think more about what you are doing, because normally I don't think very much about the steps and now you really have to write it down." (Student2, lesson 3)

Two students mentioned that by doing the steps on paper (or an online document) first, a better understanding of how to transfer steps to GeoGebra was established.

> "Once I write it down on paper, I know what the formula is I need to put in and I think about what... How does the program read the formula? And then I just fill that in." (Student16, lesson 3)

> "I think it is easier to put it (the steps) in a Word document first. Then it is easier to read it again. [...] In GeoGebra, it is sometimes not clear to me if I am doing it right" (Student8, lesson 3)

Furthermore, 5 students were able to explain their usage of conditional statements to handle special cases within GeoGebra.

> "So if x(A) equals x(B)... So if they are underneath each other... Then you do this formula, then x is just x(A), so it is a straight

> line up-down from x(A). And if that is not the case, so if they are not above each other, then we can just use the normal formula," (Student6, lesson 3)

Two students also mentioned the similarity of conditional statements within GeoGebra to other programming languages, mainly touching upon the difference in syntax that is needed to get it working.

> "What you use for that (handling special case) is actually very similar to what you do in most programming languages with conditionals. And then it's worded a bit differently, but actually it's very similar. [...] The first time you learn it, it's all new and then we went from Javascript to PHP and then you notice that it is more a matter of translating than learning something completely new." (Student1, lesson 3)

Similarly, 3 students explained that object formation in GeoGebra was not that different from other programming languages and that already having this knowledge made it easier to define variables and functions in GeoGebra.

> "No, it is actually a bit the same. Written slightly differently, but we have the fundamentals basically." (Student9, lesson 3)

Lastly, although we were limited in asking questions concerning the use of iteration commands, 3 students noted that it was rather straightforward to use these commands, and did not encounter noteworthy problems.

4.4.2 Problem solving techniques

During the lessons, students showed several strategies and techniques to solve problems. These were captured under the code problem solving. We will start with the subcode generalisation since this was one of the most important skills throughout the lessons. Next, we will explore other techniques that emerged throughout the exercises.

4.4.2.1 Generalisation

During all chapters, students started with working on a solution to a specific problem. Next, they generalized this solution to make it suitable for all instances of the particular problem. Throughout the interviews, 7 students mentioned that they did not encounter any problems with transitioning from a specific to a general solution. Two of those students responded that through the course Mathematics D, they already got in contact with the concept of generalisation, which made it easier for them.

> "I always get problems like this in Maths D, where first you get a question about how to solve it and then they generalise it so

> that you have to describe it in maths for every scenario, so to speak. So I just did that a bit intuitively and it went well, in a way." (Student 14, lesson 3)

Furthermore, one student remarked that the prior programming knowledge of the class might have some influence on the understanding of generalisation.

> "Of course, we have had computer science for quite a long time, so we can really skip the first step, because we already know that this is the same as using a variable in JavaScript." (Student9, lesson 3)

Another student mentioned that looking at things from a general perspective was something that was already ingrained in the way she was thinking, which made the transition intuitive for her.

> "I actually found it quite doable... I think that perhaps in general, I just think a bit more generally about it, about how it all fits together. Because yes... that may have something to do with the way I think. [...] I did not have the feeling that I had to think about things, they were just answers that I already had ready, actually." (Student1, lesson 3)

She also remarked that this actually resulted in her transitioning from a general to a specific solution, or skipping the entire specific solution. Four other students also explained that this happened to them, where one student also elaborated on that he does not see the need for a specific solution.

> "If I have a specific problem, I first look at how I can solve it in general and then for the specific case... So that's exactly the opposite of what we are actually doing now." (Student1, lesson 4)

> "I don't think it would have been necessary to do the first part (specific solution) if we already knew how to do the general part. I think we already knew how the formulas worked." (Student7, lesson 4)

Contrarily, 3 students expressed that they preferred only working with the specific solution, due to longer formulas that can become unclear to them.

> "Then I do not use that whole long formula with $a+b$ you know. Then I just do that normal formula, and then I just change those numbers." (Student4, lesson 4)

> "If you have to do $a + b + x$ of a and x of b very often, then it takes a lot of time and you can make mistakes pretty quickly." (Student3, lesson 4)

Lastly, some students (3 students) also expressed thought about the benefits of using generalisation and having a general solution. These mainly consisted of having an example ready for getting to the general solution, as well as adaptability and applicability to similar problems.

> "Yes, I thought it was nice to start with a specific case, because then you understand what it is about and what it should look like in the end." (Student19, Lesson 3)

> "Then you can apply it to any... For example, for a triangle you can... For any triangle you can find the midpoint of one of those lines... Always." (Student8, lesson 3)

4.4.2.2 Other techniques

Although our focus mostly was on generalisation, we also observed that students elaborated other problem solving techniques. Firstly, one student noted that during a more challenging task she used decomposition to break up the problem into more manageable pieces.

> "With that more difficult problem, it was necessary to look at the steps to see if you could make some progress, because if you don't get to the end result, at least you have done some of the work and put in the effort." (Student11, lesson 4)

Furthermore, three students mentioned that they had re-used certain parts for answers in later paragraphs. This mainly consisted of formulas for lines and conditional statements.

> "I just reused those if-then functions and adjusted them a bit when necessary" (Student7, lesson 3)

Lastly, students also seemed to share their experiences in evaluating their answers within GeoGebra. Six students mentioned that by making use of visual aspects like dragging points across the screen they were able to check if their answers were correct.

> "See, we can drag these points, the line keeps moving... So with that, you could sort of check that the general formula works." (Student19, lesson 3)

4.4.3 Difficulties during the lessons

The majority of the students (13 students) that were interviewed explicitly stated that they did not encounter any problems with the calculus content during the lessons. Some students (6 students) expressed that they had some issues with derivatives, conditional statements, iteration and macros. However, by asking questions to identify the source of these problems it appeared they did not stem from a lack of knowledge about the topic, but rather unfamiliarity with the syntax.

> "I found it a bit vague, I didn't quite understand it. How I should write it down exactly." (Student15, lesson 3)

A prevalent topic during the interviews was the students view on the use of GeoGebra and the difficulties that arose during the use of this tool. Firstly, a majority of students (13 students) noted that although they did not experience major problems during the lessons, the use of GeoGebra and getting familiar with its syntax took some trial and error in the early paragraphs.

> "It is just like any other computer programme. It takes some time getting used to, but once you spend some time in it, it is just easy to use." (Student16, lesson 4)

> "In the beginning it was a bit difficult, because we didn't know the syntax yet." (Student8, lesson 3)

These problems with syntax mostly manifested themselves through typos. The misuse of spaces and brackets, confusing plus and minus signs, and sudden creation of fractions mainly caused students to struggle within the GeoGebra environment.

> "I think just like everyone else, I still make those really stupid mistakes with pluses and minuses now and again. That I just forget a plus or minus and then I sit there for an hour staring at it like 'why doesn't it work? What have I done wrong?' But then it turns out that I just confused a plus and a minus somewhere and then it is all just fine." (Student1, lesson 4)

Another student mentioned that such tiny mistakes or typos could result in their files not working properly,

> "You just need to make sure you don't make any typing mistakes. So that... Because one mistake makes the whole... Makes everything not work." (Student16, lesson 3)

Moreover, some students (4 students) also experienced some frustrations while working with GeoGebra. The cause of this frustration was because of putting in long formulas, or the in their eyes lack of correspondence between what to write down in the workbook and what to put into GeoGebra.

> "Sometimes it takes longer to put it in the program itself than it does to sort of figure out how to do it." (Student4, lesson 3)

Furthermore, one student mentioned that some functionalities of GeoGebra did not work that well for him regarding function alteration. He also mentioned that at some point, a panel that was part of GeoGebra vanished which he could not retrieve. Another student demonstrated that using the English version of the software interfered with the workbook, which was in Dutch. This resulted in a difference of the 'ALS-DAN-ANDERS' command which was used for conditional statements.

Contrary to these issues, five students also acknowledged some benefits of using GeoGebra which mainly touched upon the usefulness of the visualisation that GeoGebra offers. Furthermore, another student noted that he preferred the use of GeoGebra over his graphic calculator.

> "I think it is a useful programme in itself, because then of course... If you fill in a formula... It's easier to visualise." (Student2, lesson)

> "I prefer GeoGebra over like a graphic calculator, because it is just easier put in that way." (Student6, lesson)

Chapter 5

Discussion

Within this chapter, we will first summarize the main findings from the previous chapter. Furthermore, we will discuss and interpret our results to aid in answering our research questions. Furthermore, we will also elaborate on the implications of these results and linking them to existing literature. Lastly, we will discuss the limitations of our research, and give recommendations for future works.

5.1 Summary of results

Firstly, the results from the pre- and post-test indicated that students initially already performed quite well on several computational tasks concerning algorithmic thinking, evaluation, repetition structures and pattern recognition. Scores between pre- and post-test developed positively or stayed rather stable. Especially a question about algorithmic thinking and evaluation saw a stark increase in positive scores. However, when students were confronted with questions that asked to apply their acquired GeoGebra knowledge to new situations they seemed to struggle significantly more.

Looking at the data from the workbooks, we found that students mainly struggled in the early paragraphs. This was mainly due to students struggling to come up with well-formed formulas and conditional statements. We observed that these mistakes with conditional statements were not due to a lack of understanding about handling special cases, but rather a consequence of wrongfully defining formulas for e.g. lines. Apart from these difficulties, we noticed that students were adequate in solving calculus tasks that involved the use of algorithmic thinking and generalisation skills in the workbook, which were reflected in higher scores as the lessons progressed.

From the analysis of the GeoGebra files, we observed that students were adequate in performing the exercises within GeoGebra. However, similarly to the workbooks, students mainly made mistakes with defining mathematically correct formulas. This resulted in chain-reactions within the files where

e.g. conditional statements would be incorrect. A last interesting finding was the lack of object formation that would contribute to help students with conciseness in their files.

We also analysed the hand-in rates for the workbooks and GeoGebra files. Especially for the GeoGebra files, we saw disappointing individual hand-in rates. We acknowledged that students often work together in one file and hand it in, and as such we also checked the group hand-in rates. Looking at a similar experiment [45] we noticed that the individual rates for handing in the workbooks was satisfactory (65% opposed to 67%). However, for the GeoGebra files we observed that the hand-in rates were not satisfactory (50% opposed to 67%). Additionally, the group hand-in rates for these files resulted in only 55% of groups handing in their files which still was not satisfactory.

Lastly, from the interviews we were able to observe student strategies regarding algorithmic thinking and generalisation, as well as their experiences with GeoGebra. Having interviewed 19 of 20 students at least once, we noticed that students were capable of expressing thoughts about the use of steps, conditional statements and the benefits of generalisation. Furthermore, students communicated that they welcomed the visualisation aspect that GeoGebra offered, after overcoming the initial hurdles of getting familiar with the syntax.

5.2 Triangulation and interpretation

Our objective for this thesis was to answer the following research question and respective subquestions:

MRQ: *What characterises the use of algorithmic thinking and generalisation skills by 16-17 year old secondary school students after a 5-lesson calculus course that focused on the use of the mathematical tool GeoGebra?*

SRQ1: *In which data sources is algorithmic thinking and generalisation apparent?*

SRQ2: *What are the learning outcomes of students regarding algorithmic thinking and generalisation skills after a 5-lesson calculus course that focused on the use of the mathematical tool GeoGebra?*

SRQ3: *What challenges did students encounter during the learning process?*

In the following subsections, we will interpret our results and make an effort in answering these research questions.

5.2.1 Evaluation of data sources

Regarding our first sub-question, we focus on the different assessment methods that have been used throughout the lesson series. With the results of our analysis, we observed that all data sources were able to capture the use of AT and generalisation skills in their own way.

In the pre- and post-test, AT was mainly employed through questions requiring construction and evaluation of algorithms. We could then analyse these answers to capture and evaluate the use of AT. Furthermore, through the analysis of workbooks and GeoGebra files, we were able to capture student performance with tasks that required students to translate steps in the workbook to computational steps in GeoGebra, use logical structures such as conditional statements, and generalise from specific to general solutions. In combination with student interviews in which we could ask students about their experiences and strategies in solving these tasks, we were able to come up with statements regarding the use of AT and generalisation.

Additionally, we would like to add that by using a variety of data sources, it enabled us to triangulate our data to discover patterns that initially may not have been exposed by only analyzing a single data source. This is in accordance with Grover [23], who noted that finding comprehensive statements about computational thinking skills development needs multiple, differing assessment methods. Using these methods on a standalone basis would not suffice in giving grounded statements. For example, quotes from the interviews would serve no foundation for statements without being able to connect them to actions within GeoGebra or workbooks. Furthermore, we would like to add that apart from the pre- and post-test, the assessment methods that were used were made exclusively for this particular course. Although it was able to capture the usage of AT and generalisation for this course, does not guarantee that these methods also are adequate for other activities regarding AT and generalisation skills.

5.2.2 Learning outcomes

In relation to our second sub-question, we observed that students were successful in solving the problems that required the employment of AT and generalisation skills within the workbook and GeoGebra. Students showed to be capable of constructing algorithms within the GeoGebra environment, from procedures that were created in the workbook. Additionally, during the interviews students were able to correctly explain their actions regarding setting up procedures and the use of logical structures. Remarkably, we noticed that students who made mistakes within the workbook, often corrected these mistakes when handing in the accompanied GeoGebra file. This shows that students learned to correctly evaluate their own algorithms within GeoGebra. This is an interesting finding since students initially scored rather

poorly on evaluating algorithms in the pre-test, whereas these scores significantly increased in the post-test. A possible explanation could be that students were able to validate their answers through the visualisation aspect that GeoGebra offers. In the literature, researchers stress the importance of visualisation as a way to support the use of AT in students that struggle with it [2, 20].

Concerning generalisation, we observed that students were able to generalize from a specific to a general solution. Furthermore, in the interviews students were able to express ideas about the benefits of generalisation, such as the applicability of solutions to new situations. Some students already showed a good understanding of generalisation since they often did not require a specific solution. In our view, this can be attributed to the background of the participants, who already have prior knowledge in programming which makes them familiar with the use of variables. Moreover, some students followed a mathematics course in which they already got in contact with generalisation. This also could have made the tasks of this lesson series more intuitive.

Lastly, we observed that students were able to come up with general plans of action when confronted with more challenging tasks. This mainly showed through the last paragraph of the workbook and the post-test specific questions. Especially during the last paragraph, students were confronted with new a new (mathematical) topic, while still tapping into the acquired basics of programming in GeoGebra. Although students struggled slightly with becoming concrete, we still consider this outcome as beneficial. This transfer of GeoGebra knowledge to different contexts is considered an important part of learning since it shows a good understanding of the relevance of their knowledge [5]. This transfer could be related to generalisation since it is considered as transferring knowledge from a specific setting to another one as was emphasised by Dumitrascu [18]. Overall, we conclude that student learning outcomes consist of (1) successfully constructing and evaluating algorithms while using logical structures within GeoGebra, (2) generalising specific solutions into general solutions which apply to every instance of the problem and (3) the ability to transfer knowledge of GeoGebra to more challenging problems. In our view, these outcomes are satisfactory considering that they line up with the goals of the course.

5.2.3 Challenges during the learning process

During the course, we noticed that students did not encounter major challenges. We did notice some difficulties in the first 3 paragraphs of the course which was apparent from the workbooks. These paragraphs concerned defining lines, perpendicular lines, medians and centres of gravity. The second part (paragraphs 4 to 6) concerned tangents and derivatives of standard functions. In this part we did not encounter any 'Inadequate' scores any-

more. This suggests that students were more adequate with content related to derivatives and tangents. We acknowledge that it is difficult to determine if this relative switch in content is the cause of better scores later on, since students did not discuss any significant difficulties about the content during the interviews.

However, students expressed that they were affected by the initial unfamiliarity with the syntax of GeoGebra. Nonetheless, within the GeoGebra files we rarely encountered syntax that did not make sense. This can again be attributed to the prior programming knowledge of the students. This was also observed from the interviews, in which students noted that e.g. conditional statements mostly coincided with other programming languages and as such were easier to get familiar with. From a paper by van Borkulo et al. [45], which carried out the same calculus course at another school, it became apparent that students struggled significantly more with these logical structures. In their research, students did not have prior experience with programming. Research has shown that students benefit from having prior programming experience, and often perform better than those who have none when introduced to new languages or syntax [24, 31]. In our view, this explains the difference between these two groups of students, and our low number of syntax mistakes within the GeoGebra files.

Within the workbooks, students mainly struggled to come up with general solutions. However, they did not mention these difficulties during the interviews. This is remarkable, but also explainable through another challenge that students encountered and has lead to the majority of mistakes within the workbooks and GeoGebra. This challenge was the miscalculation of formulas, which stems from a misunderstanding of the mathematical content. We observed that mistakes of this kind resulted in a chain reaction of mistakes throughout the exercises since e.g. parts of the general solution were used within the definition of the conditional statements. It remains unclear why students made these mistakes while also acknowledging no difficulties with the calculus content of the course. A possible explanation for this could be the fact that the workbooks were designed to be used with pen and paper. Students however filled in the workbooks within text editors, which takes away this pen and paper aspect. Interestingly the results from van Borkulo et al, showed that students made fewer mistakes concerning formulas [45]. Contrarily to the participants of our research, the students in their research were able to make use of physical workbooks. This finding suggests that the use of pen and paper to come up with steps and formulas is more beneficial than using (online) text editors.

5.2.4 Characterization

Ultimately, how can the use of algorithmic thinking and generalisation skills by students be characterized after the course? In our opinion, we can con-

clude that the course itself is very suitable to bring students in contact with concepts of calculus in a refreshing manner. Students welcomed the use of GeoGebra and our findings show that students were adequate in solving the problems in the workbooks and GeoGebra using algorithmic thinking and generalisation skills. Students showed that they were able to generalise from specific to general solutions, translate step-by-step procedures into algorithms within GeoGebra, use logical structures to handle special cases, and evaluate algorithms using visualisation features of GeoGebra. Another characteristic was the ability of students to transfer their acquired GeoGebra knowledge to new, more challenging situations. These findings coincide with literature on abilities that can be associated with algorithmic thinking [11, 19, 29] and generalisation [18, 41].

Throughout the course, students did not encounter major hurdles while familiarizing themselves with the digital environment GeoGebra and its syntax. Through prior knowledge of programming, students expressed that they did not struggle with defining variables and using logical structures, which was reflected in the GeoGebra files. In the case where mistakes were made regarding e.g. conditional statements, we observed that they did not originate from a misunderstanding of the concept itself, but rather an accumulation of mistakes in defining mathematically correct formulas. Remarkably, in the interviews students overwhelmingly expressed that they did not encounter problems with the calculus content. We believe that this contradiction and the mathematical mistakes could be attributed to the absence of a physical workbook. Research has shown students who type certain science-related content in a digital environment are more likely to get cognitively overloaded than students who write using pen and paper [1]. They acknowledge that because of this, students show *"less knowledge, less terminological accuracy, and, above all, less understanding of the interconnection between listed information."*.

5.3 Limitations

One of the main limitations of our research was the special circumstance in which the course was held. Due to the COVID-19 pandemic, students were required to follow the course from their homes. Research shows that this form of online education is not for everybody [8, 30]. Some students simply prefer a classroom setting where a teacher is nearby to help with problems and to keep them motivated to study. Furthermore, online courses usually require proper design of materials and a thorough selection of students. For the design of this course, these unfortunate circumstances were not taken into account which in our view resulted in students being less engaged in the course. This was mainly reflected in the disappointing individual hand-in rates for the workbooks and GeoGebra files, which significantly shrunk

usable data for our analysis.

Furthermore, another limitation was the small number of participants in this research. Our target audience covers a larger number, while also differentiating between levels of prior knowledge. Although the findings in our study are backed up by our analysis, and mostly coincide with findings from a similar paper on the implementation of this course [45], we acknowledge that the generalizability of these findings could be compromised. This also relates to the previously mentioned limitation, since the circumstances under which this research has taken place were different from teaching pre-pandemic.

Lastly, we would like to address a limitation related to one of our assessment methods. Although the interviews proved to be fruitful, we observed that conducting interviews in a group setting undermined the potential for deeper insights due to longer speaking times for certain students. Furthermore, a complication that can occur during the analysis is researcher bias while interpreting the data [16], whereas with one-on-one interviews this is less likely. However, we are aware of the time-consuming aspect of doing individual interviews which was one of the main consideration in opting for group interviews for this research. Nonetheless, we think that addressing this limitation is needed, since we observed that some students who got interviewed were not able to fully express their experiences, strategies and challenges over the course of the lesson series.

5.4 Future work

For future endeavours, research might focus on the role of the teacher throughout the course. Although in our research the teacher had familiarity with both teaching mathematics and computer science, in future implementations this might not be the case. It was beyond the scope of this thesis to investigate the role of the teacher, but the challenge in the future is to train and guide teachers in addressing CT during their lessons while also supporting students in acquiring and using skills related to CT.

Furthermore, another interesting topic for future research could be the impact of plugged versus unplugged learning on algorithmic thinking skills development. Within our research, we found that students almost entirely did all the exercises within a digital environment, albeit a text editor or GeoGebra. It would be interesting to contrast the results of students that worked out steps on paper, which was the goal of the workbook in our research, and students that worked out steps within a digital environment. Similarly, research into the differences in employment of algorithmic thinking and generalisation skills between groups that have had prior experience in programming to those who have none could also be an interesting topic. This indirectly suggests that research into this topic should be done on a

larger scale, which follows from one of the limitations of our research.

Lastly, it would be beneficial for the field to not only characterize the use of algorithmic thinking and generalisation skills but also investigate the development of these skills. This requires adequate assessment tools, of which the designed pre- and post-test functions as a good starting point. Furthermore, interviews that are more artefact-based and held one-on-one could also aid in catching this development [10].

Chapter 6

Conclusion

Incorporating computational thinking within STEAM education has received increased attention over the last decade. Computational thinking is still an ill-defined construct, which complicates the assessment of usage and development of associated elements. In this thesis, we tried to characterize the use of two elements of CT, algorithmic thinking and generalisation, after a 5-lesson calculus course that revolved around the use of the mathematical tool GeoGebra.

This thesis has shown that GeoGebra can be employed as a tool to provide students with calculus content in a refreshing manner, while also touching upon aspects of CT. Analysis of pre- and post-tests, student work in workbooks and GeoGebra, and mini-interviews with students demonstrated that students welcomed the use of GeoGebra and were adequate in solving tasks that required the use of algorithmic thinking and generalisation skills, such as constructing and evaluating algorithms from solutions that were acquired through generalisation. Although students needed time to familiarize themselves with the syntax and digital environment, they did not encounter major challenges regarding algorithmic thinking and generalisation. Findings showed that the majority of mistakes that were made during the course were due to mathematical shortcomings.

For future research, the role and training of teachers in leading these kinds of courses could be investigated to provide insight into their position as a means to stimulate the usage and development of skills related to CT. Furthermore, larger-scaled experiments could focus on the differences in algorithmic thinking skills development between plugged and unplugged activities, and the contrast between groups that have prior knowledge in programming in comparison to those who have none. Lastly, rather than only characterizing the usage of algorithmic and generalisation skills, by investigating and developing better assessment methods we might also gain insight into how these skills develop over time during similar integration efforts.

Bibliography

[1] Metka Kordigel Aberšek, Boris Aberšek, and Andrej Flogie. Writing versus typing during science teaching: case study in slovenia. *Journal of Baltic science education*, 17(1):84, 2018.

[2] Folake M. Adelabu, Moses Makgato, and Manto S. Ramaligela. The importance of Dynamic Geometry Computer Software on learners' performance in geometry. *Electronic Journal of E-Learning*, 17(1):52–63, 2019.

[3] Cláudio Amorim. Beyond Algorithmic Thinking: An Old New Challenge for Science Education. In *Eighth International History, Philosophy, Sociology & Science Teaching Conference, July*, volume 15, pages 1–9, 2005.

[4] Thiago S. Barcelos and Ismar F. Silveira. Teaching computational thinking in initial series an analysis of the confluence among mathematics and computer sciences in elementary education and its implications for higher education. In *2012 XXXVIII Conferencia Latinoamericana En Informatica (CLEI)*, pages 1–8. IEEE, 2012.

[5] Susan M. Barnett and Stephen J. Ceci. When and where do we apply what we learn?: A taxonomy for far transfer. *Psychological bulletin*, 128(4):612, 2002.

[6] Valerie Barr and Chris Stephenson. Bringing computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community? *Acm Inroads*, 2(1):48–54, 2011.

[7] Tim Bell and Michael Lodi. Constructing computational thinking without using computers. *Constructivist foundations*, 14(3):342–351, 2019.

[8] Joe Bocchi, Jacqueline K. Eastman, and Cathy Owens Swift. Retaining the online learner: Profile of students in an online MBA program and implications for teaching them. *Journal of education for Business*, 79(4):245–253, 2004.

[9] Sinead Breen and Ann O'Shea. Mathematical thinking and task design. *Irish Mathematical Society Bulletin*, (66):39–49, 2010.

[10] Karen Brennan and Mitchel 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*, volume 1, page 25, 2012.

[11] Stephen Cooper, Wanda Dann, and Randy Pausch. Developing algorithmic thinking with Alice. In *The proceedings of ISECON*, volume 17, pages 506–539. Citeseer, 2000.

[12] Andrew Csizmadia, Paul Curzon, Mark Dorling, Simon Humphreys, Thomas Ng, Cynthia Selby, and John Woollard. Computational thinking - A guide for teachers. 2015.

[13] Computer Science Teacher Association, (CSTA) and International Society for Technology in Education (ISTE). Operational definition of computational thinking for K-12 education., 2011, [Online]. Retrieved from: http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf.

[14] Commissie Toekomst Wiskundeonderwijs (cTWO). Denken & doen, wiskunde op havo en vwo per 2015, Utrecht: cTWO, 2013. Retrieved from: http://www.fi.uu.nl/ctwo/publicaties/docs/CTWO-Eindrapport.pdf.

[15] Keith J. Devlin. *Introduction to mathematical thinking*. Keith Devlin Palo Alto, CA, 2012.

[16] Owen Doody, Eamonn Slevin, and Laurence Taggart. Focus group interviews part 3: Analysis. *British journal of nursing*, 22(5):266–269, 2013.

[17] Paul H. M. Drijvers. Denken over wiskunde, onderwijs en ICT: Inaugurale rede, *Utrecht University*. 2015.

[18] Gabriela Dumitrascu. Understanding the process of generalization in mathematics through activity theory. *International Journal of Learning, Teaching and Educational Research*, 16(12):46–69, 2017.

[19] Gerald Futschek. Algorithmic thinking: the key for understanding computer science. In *International conference on informatics in secondary schools-evolution and perspectives*, pages 159–168. Springer, 2006.

[20] Gerald Futschek and Julia Moschitz. Learning algorithmic thinking with tangible objects eases transition to computer programming. In *International conference on informatics in schools: Situation, evolution, and perspectives*, pages 155–164. Springer, 2011.

[21] GeoGebra. About GeoGebra, n.d., [Online]. Retrieved from: https://www.geogebra.org/about.

[22] Lindsey A. Gouws, Karen Bradshaw, and Peter Wentworth. Computational thinking in educational activities: an evaluation of the educational game light-bot. In *ITiCSE '13*, 2013.

[23] Shuchi Grover, Roy Pea, and Steve Cooper. Systems of assessments" for deeper learning of computational thinking in K-12. In *Proceedings of the 2015 annual meeting of the American educational research association*, pages 15–20, 2015.

[24] Dianne Hagan and Selby Markham. Does it help to have some programming experience before beginning a computing degree program? In *Proceedings of the 5th annual Conference on Innovation and Technology in Computer Science Education (CITCSE)*, pages 25–28, 2000.

[25] Štepan Hubálovskỳ and Ondrej Kořínek. Evaluation of algorithmic thinking of students using control testing environment. *Int. J. Educ. Inf. Technol. North Atlantic University Union*, pages 2074–1316, 2015.

[26] Nicole M. Hutchins, Gautam Biswas, Miklós Maróti, Ákos Lédeczi, Shuchi Grover, Rachel Wolf, Kristen P. Blair, Doris Chin, Luke Conlin, Satabdi Basu, et al. C2STEM: A system for synergistic learning of physics and computational thinking. *Journal of Science Education and Technology*, 29(1):83–100, 2020.

[27] Filiz Kalelioglu, Yasemin Gulbahar, and Volkan Kukul. A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 4(3):583–596, 2016.

[28] Maria Kallia, Sylvia P. van Borkulo, Paul Drijvers, Erik Barendsen, and Jos Tolboom. Characterising computational thinking in mathematics education: a literature-informed delphi study. *Research in Mathematics Education*, pages 1–29, 2021.

[29] Zoltán Kátai. The challenge of promoting algorithmic thinking of both sciences-and humanities-oriented learners. *Journal of Computer Assisted Learning*, 31(4):287–299, 2015.

[30] Greg Kearsley. Is online learning for everybody? *Educational technology*, 42(1):41–44, 2002.

[31] Zoe A. Kersteen, Marcia C. Linn, Michael Clancy, and Curtis Hardyck. Previous experience and the learning of computer programming: The computer helps those who help themselves. *Journal of Educational Computing Research*, 4(3):321–333, 1988.

[32] Ung L. Ling, Tammie C. Saibin, Jane Labadin, and Norazila A. Aziz. Preliminary investigation: Teachers' perception on computational thinking concepts. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 9(2-9):23–29, 2017.

[33] John Mason. Expressing generality and roots of algebra. In *Approaches to algebra*, pages 65–86. Springer, 1996.

[34] Massachusetts Institute of Technology. About Scratch, n.d., [Online]. Retrieved from: https://scratch.mit.edu/about.

[35] Chrissy Monteleone, Paul White, and Vince Geiger. Defining the Characteristics of Critical Mathematical Thinking. *Mathematics Education Research Group of Australasia*, 2018.

[36] Andreas Mühling, Alexander Ruf, and Peter Hubwieser. Design and first results of a psychometric test for measuring basic programming abilities. In *Proceedings of the workshop in primary and secondary computing education*, pages 2–10, 2015.

[37] George Pólya. On learning, teaching, and learning teaching. *The American Mathematical Monthly*, 70(6):605–619, 1963.

[38] Marcos Román-González. Computational thinking test: Design guidelines and content validation. In *Proceedings of EDULEARN15 conference*, pages 2436–2444, 2015.

[39] Marcos Román-González, Juan-Carlos Pérez-González, and Carmen Jiménez-Fernández. Which Cognitive Abilities Underlie Computational Thinking? Criterion Validity of the Computational Thinking Test. *Computers in Human Behavior*, 72(C):678–691, July 2017. doi:10.1016/j.chb.2016.08.047.

[40] Alan H. Schoenfeld. Learning to think mathematically: Problem solving, metacognition, and sense making in mathematics (Reprint). *Journal of Education*, 196(2):1–38, 2016.

[41] Cynthia Selby and John Woollard. Computational thinking: the developing definition, University of Southampton, 2013. Retrieved from: https://eprints.soton.ac.uk/356481/.

[42] Cary Sneider, Chris Stephenson, Bruce Schafer, and Larry Flick. Exploring the science framework and NGSS: Computational thinking in the science classroom. *Science Scope*, 38(3):10, 2014.

[43] Kaye Stacey. What is mathematical thinking and why is it important, 2006. Retrieved from: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.550.1647&rep=rep1&type=pdf.

[44] David Tall. The nature of advanced mathematical thinking. In *Proceedings of Psychology of Mathematics Education Conference, Hungary*. Citeseer, 1988.

[45] Sylvia van Borkulo, Christos Chytas, Paul Drijvers, Erik Barendsen, and Jos Tolboom. Computational Thinking in the Mathematics Classroom: Fostering Algorithmic Thinking and Generalisation Skills Using Dynamic Mathematics Software. To appear in *Proceedings of the 16th Workshop in Primary and Secondary Computing Education (WiPSCE)*. 2021.

[46] Wesley Vernon. The Delphi technique: a review. *International Journal of Therapy and rehabilitation*, 16(2):69–76, 2009.

[47] Kevin P. Waterman, Lynn Goldsmith, and Marian Pasquale. Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, 29(1):53–64, 2020.

[48] Joshua L. Weese and Russell Feldhausen. STEM outreach: Assessing computational thinking and problem solving. In *ASEE Annual Conference & Exposition*, 2017.

[49] David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, and Uri Wilensky. Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1):127–147, 2016.

[50] David Weintrop, Shandra Morehouse, and Mega Subramaniam. Assessing computational thinking in libraries. *Computer Science Education*, 31(2):290–311, 2021.

[51] David Weintrop and Uri Wilensky. Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In *ICER*, volume 15, pages 101–110, 2015.

[52] David Weintrop, Daisy Wise Rutstein, Marie Bienkowski, and Steven McGee. Assessing computational thinking: an overview of the field, 2021.

[53] Jeannette M. Wing. Computational thinking. *Communications of the ACM*, 49(3):33–35, 2006.

[54] Jeannette M. Wing. Research Notebook: Computational thinking - What and Why? The link. *The magazine of the Carnegie Mellon University School of Computer Science*,

2011. [Online]. Retrieved from: http://www.cs.cmu.edu/link/research-notebookcomputational-thinking-what-and-why.

Appendix A

Links to content, raw data and analysis results

Workbook and pre-/post-tests

Pre- and post-test:
https://docs.google.com/document/d/1FIdWwWzLmzAlpKLUvr9tdtrGT8bZyZ-4cUrB3wlqZhY/edit?usp=sharing

Workbook:
https://docs.google.com/document/d/1CnpW-M2TdwOooRVDKi3QyTMJaDvII3Zl/edit?usp=sharing&ouid=117080045490976004325&rtpof=true&sd=true

Raw data/analysis results

Pre- and post-test:
https://docs.google.com/document/d/1xizJziLhxNFvM27zOLevZE5DBbGNkQxzwbc4deM2uc4/edit?usp=sharing

Workbooks:
https://docs.google.com/document/d/1agEwWRLIKL1tsbN6rxeuiE0HcAtKhGqxHE98dd0nJPc/edit?usp=sharing

GeoGebra files:
https://docs.google.com/document/d/1jnmVYRj0lB1LADjKmKWB5fFWjoCU37MyGbVQRZbzZOw/edit?usp=sharing

Interview transcriptions:
https://docs.google.com/document/d/1yhd8hycxIympzgQmTrRGbygDAu7KjFBFmFQVNMKOg6E/edit?usp=sharing

Appendix B

Codebook for interview analysis

Code group	Code	Subcode	Subcode Description
CMT aspects	Algorithmic thinking	Steps	This code is used when a student expresses thoughts about using algorithms, setting up algorithms, and when a student refers to step by step procedures, or defining steps.
		Conditional statements	This code is used when a student refers to construction if-then-else statements, including exceptions.
		Object formation	This code is used when a student refers to the formation or the use of objects, such as created functions or variables in GeoGebra.
		Loops/Iteration	This code is used when a student refers to the construction of iteration commands/loop commands.
	Problem solving	Decomposition	This code is used when a student refers to breaking up a problem, using smaller parts or steps for better understanding. When only 'steps' are referred to, this is coded by 'Algorithmic thinking – steps'.
		Generalisation	This code is used when a student refers to a general solution, including 'first specific then general'.
		Evaluation	This code is used when a student expresses thinking of what is a correct answer, or how GeoGebra can be used to check the solution, see that it works.

		Strategies	This code is used when a student describes approaches of having an overview, reusing a solution, trying out in GeoGebra, using CT and a step by step approach to solve problems in general, keeping calm.
Tools & technology	Tools	GeoGebra	This code is used when GeoGebra is mentioned. For example, the combination of codes 'GeoGebra' and 'difficult/unclear' marks a perception of the tool Geogebra.
		Workbook	This code is used when the workbook is mentioned. For example, the combination of codes 'Workbook' and 'easy/clear' marks a perception of the workbook.
		Other tools	This code is used when other tools are mentioned, for example the graphical calculator or the use of pen and paper.
		(Dis)advantages medium	This code is used when a student refers to advantages or disadvantages of the medium, for example how the tool can be a vehicle for understanding, or how the tool hinders understanding.
		Knowledge, skills	This code is used when a student mentions knowledge or skills related to technology, for example, when talking about know how the tools works, know how to use tool, tool experience.

	Technology	Technical features	This code is used when technical features are mentioned or used, such as copy & paste, dragging objects, formulas, commands, functions, sliders, variables, visualisations.
		Technical issues	This code is used when technical issues are mentioned or encountered, such as syntax, notation, things changing automatically, save problems.
Descriptives	Students	Student1, ... StudentX	The subcodes indicate the students who are interviewed.
	Progress materials	Chapter1, ... ChapterX	The subcodes indicate the progress in the materials or the part about which is spoken, for example 'Chapter 2' or 'Chapter 2.3'.
	Maths content	Parabola, tangent line, ...	The subcodes indicate the mathematical content, for example 'tangent line', or an aspect in general related to the mathematics lessons, for example, 'good to repeat math'.
	Student perception	easy, difficult, handy, frustration, ...	This code group contains codes that represent a 'quality' or 'qualification' that can be used for another topic, for example 'easy, clear, doing fine' for 'Chapter 2.3', or 'difficult, unclear' for 'Geogebra'. This group also includes topics about 'the way of working' such as handling frustration, trial and error, working from scratch, being precise, having fun, asking help from teacher.