

Supporting Students' Computational Thinking and Mathematical Thinking in the Mathematics
Classroom

Ivana den Hartogh

5578930

30 ECTS

Supervisor: Paul Drijvers

Second supervisor: Sylvia van Borkulo

Second corrector: Wouter van Joolingen

Location: The Rijnlands Lyceum, Oegstgeest, and Christelijk Gymnasium Sorghvliet, Den Haag

Utrecht University

Written according to the guidelines of the journal *Mathematical Thinking and Learning*

Abstract

Both computational thinking (CT) and mathematical thinking (MT) have become popular in recent years and both have been recognized as essential skills for all students, as future members of this software-driven world. However, little is known about how to support students developing CT in combination with MT skills in the mathematics classroom. It is assumed that CT and MT are linked through shared skills, among which algorithmic thinking. In this design-oriented research, we explored how to support 16-17-year old pre-university students in developing CT and MT skills, by means of algorithmic thinking. We designed a 5-hour series of learning activities, to address four subskills of algorithmic thinking: (a) the ability to read an algorithm, (b) the ability to specify a problem, (c) the ability to construct an algorithm, and (d) the ability to understand an algorithm. The activities were piloted in two groups, involving 53 students in total. The data were gathered through mini-interviews and students' written work. Results showed an increase of three of the four subskills of algorithmic thinking in the written documents over the course of the pilot. Moreover, the results of the mini-interviews showed that most students were able to express a general definition of algorithmic thinking. Therefore, the results from the study suggest that an AT-oriented teaching sequence has the potential of supporting students in developing both CT and MT skills.

Keywords: Algorithmic thinking, computational thinking, design research, mathematical thinking

SUPPORTING STUDENTS' CT AND MT IN MATH

We live in a software-driven world (Manovich, 2013), and computers influence almost every aspect of our lives these days. This software-driven world comes with complex social and scientific issues, as the amount of data, variables and computing power needed increases. Examples of such issues are given by Wing (2008): robotic surgery in medicine to proving the four-color theorem in mathematics. The complexity of these issues is what makes them unsolvable without the use of computer technology. This constitutes a societal problem. To solve this problem, people need to learn how to work with a computer to make the computer solve complex social and scientific issues, and how to deal with large amounts of data. These skills are termed as Computational thinking (CT) skills. As Bower and Falkner (2015) stated, society needs to ensure that the educational systems provide not only the fundamentals of digital literacy but also the CT processes needed to understand the scientific practices that underpin technology. This introduces an educational challenge. Hence, the relevance of this research is stated by the fact that learning to think computationally should be given a place in education.

So, if the focus of the content of education has to be shifted onto teaching young people CT skills, we need to know what exactly is meant by CT. Despite of the increasing interest, there still exists a lack of consensus on a formal definition of computational thinking (Barr & Stephenson, 2011; Gouws, Bradshaw & Wentworth, 2013; Grover & Pea, 2013). Literature is far from either explaining what CT is and how to teach and assess this skill (Brennan & Resnick, 2012; Kalelioglu, Gülbahar & Kukul, 2016). Consequently, there exists a knowledge gap with regards to the teaching of computational thinking skills to students.

What we do know about CT is that it is based on *computing*. According to Wing (2008), a key voice in the field, CT is taking an approach to solving problems, designing systems and understanding human behaviour that draws on concepts fundamental to computing. In other words: CT skills represent teamwork between human and computer. It is also known that CT skills inherently draw on mathematical thinking (MT), given that, like all sciences, its formal foundations rest on mathematics (Wing, 2006). More explicitly: CT focuses on the process of abstraction, which is one of the three core elements of mathematical thinking, according to Drijvers (2015a). From a pedagogical perspective the thoughtful use of computational tools and skills can deepen the learning of mathematics and science content (Eisenberg, 2002; Repenning et al., 2010; Sengupta et al., 2013; Sherin, 2001; Wilensky et al., 2014; Wilensky & Reisman, 2006). Another argument

SUPPORTING STUDENTS' CT AND MT IN MATH

for connecting MT skills to CT skills is stated by the fact that the new curriculum for pre-university upper secondary mathematics education stresses the importance of MT as a learning goal.

Since new learning goals concerning both CT and MT skills need to be achieved, there is a need for learning activities that can help students develop those skills. One of the common elements that CT and MT are based on is algorithmic thinking (AT). In this research, we will focus on algorithmic thinking as shared skill of CT and MT. Therefore, the aim of this research is to explore how students can learn to think both computationally and mathematically, through algorithmic thinking, by engaging in learning activities. These learning activities were designed, tested, and analysed using a codebook. As a result of the explorative character of this research, the findings will indicate ideas for future research. These ideas are given at the end of this paper.

This research was a pilot study in the three-year research project: *Computational thinking and mathematical thinking: digital literacy in mathematics curricula*. For this project, a collaboration was formed among teachers, researchers and a curriculum development institute, hereafter referred to as the consortium. These teachers range in experience from 4 to 24 years. Expertise from this consortium helped me carry out this research.

Theoretical Background

Computational Thinking

It is found that literature regarding CT is at an early stage of maturity (Kalelioglu et al., 2016). Despite of the increasing interest in computational thinking, there still exists a lack of consensus on a formal definition (Barr & Stephenson, 2011; Gouws et al., 2013; Grover & Pea, 2013). As a starting point for the concept CT for this research, we used the vision of Jeannette Wing. According to Wing (2008), operationally, computing is concerned with answering ‘How would I get a computer to solve this problem?’, where the computer could be a machine, a human, the combination of a machine and a human, or recursively, the combination (e.g. a network) of such computers. Implicit in answering this question is the identification of appropriate abstractions and choosing the appropriate kind of computer for the task (Wing, 2008). From the analysis of the findings of their review study on 125 papers concerning CT, Kalelioglu et al. (2016) found that abstraction, algorithmic thinking, problem solving, pattern recognition, and design-based thinking were the top five skills underlined by researchers, and their definition of CT also consists of thinking types such as algorithmic and design-based. Thus, in this research we use the elements

SUPPORTING STUDENTS' CT AND MT IN MATH

abstraction, algorithmic thinking, problem solving, pattern recognition, algorithmic thinking and design-based thinking to depict CT. These elements are reminiscent of math education and mathematical thinking, which is why we will explore this connection further.

Mathematical Thinking

Although much attention is given to mathematical procedures to become routine actions for students in mathematics education, not these routines but insights in why procedures work are relevant for student's later life. This is what Pólya remarked long ago: ... first and foremost, it should teach those young people to THINK (Pólya, 1963). More researchers share this point of view and this had led to definitions of MT such as MT is a whole way of looking at things, of stripping them down to their numerical, structural, or logical essentials, and of analyzing the underlying patterns (Devlin, 2011, p. 59) and a key component of MT is having a disposition to looking at the world in a mathematical way, and an attitude of seeking a logical explanation (Stacey, 2006). However, these definitions are not very specific. According to Drijvers (2015b), MT is the core and the power of mathematics and it also constitutes an important intended result of mathematics education. He distinguishes three core elements of mathematical thinking: problem-solving, modelling, and abstraction. Problem solving concerns solving non-routine tasks, for which students don't have a readymade strategy available (Schoenfeld, 2007). Modelling represents the translation from realistic problems into mathematical form and backwards, and abstraction is the isolation of specific attributes of a concept so that they can be considered separately from the other attributes (Tall, 1988, p. 2). In this research we will use this characterization of skills to depict MT.

AT as shared skill of CT and MT

CT and MT share multiple aspects, such as abstraction, algorithmic thinking and problem solving, as identified before. In this research, we will focus on algorithmic thinking as shared skill of CT and MT.

The link between MT and algorithmic thinking rests on the idea of process-object duality. The process-object duality refers to mathematical concepts having both a process and an object character, the former being the most accessible one to students. To understand a mathematical concept, processes need to be objectified. This object formation is called reification (Sfard, 1991) or encapsulation (Dubinsky, 1991). Object formation also corresponds to abstraction, one of the three core elements of MT as distinguished by Drijvers (2015b). The result of the abstraction

SUPPORTING STUDENTS' CT AND MT IN MATH

process is the ability to reason on a higher level about mathematical objects and their connections. So, following an algorithm is a procedure, but explaining an algorithm (to a computer or to a person), or designing one, leads to the formation of an object. Hence, AT is a core element of MT. Therefore, in this research we will explore how pre-university students can learn to think both computationally and mathematically, through the development of algorithmic thinking as a means to create mathematical objects.

Algorithmic thinking is a core skill for constructing algorithms to solve problems and for understanding computer science (Hsu & Wang, 2018). Algorithmic thinking is an important part of CT, since programming rests on algorithms. As Castillo, Berenguer, Sánchez, and Álvarez (2017) suggested, programming is best taught using algorithms in a previous course and then move on to using a programming language.

To be able to focus on algorithmic thinking, a more precise description is needed. Several definitions can be found in literature. First, the members of the NRC committee (1999) define general concepts of algorithmic thinking as functional decomposition, repetition (iteration and/or recursion), basic data organizations, generalization and parameterization, algorithm vs. program, top-down design, and refinement. A second definition is given by Futschek (2006). He claims that algorithmic thinking is somehow a pool of abilities that are connected to constructing and understanding algorithms: the ability to analyse given problems; the ability to specify a problem precisely; the ability to find the basic actions that are adequate to the given problem; the ability to construct a correct algorithm to a given problem using the basic actions; the ability to think about all possible special and normal cases of a problem and the ability to improve the efficiency of an algorithm (Futschek, 2006). Lastly, Lamagna (2015) found that algorithmic thinking is the ability to understand, execute, evaluate, and create computational procedures. He also claims that algorithmic thinking requires the ability to evaluate procedures for both their correctness and efficiency as well. Given a task, an accomplished algorithmic thinker can develop a sequence of precise, step-by-step instructions that always solves the task completely, correctly, and efficiently (Lamagna, 2015). So, for now, our working definition of AT is: The ability to construct and understand algorithms.

In order to make AT measurable, a more precise definition of AT is needed. Therefore, we will now specify the working definition further into core abilities. The target audience of our research is not expected to know what an algorithm is. That is why we chose to include the reading

SUPPORTING STUDENTS' CT AND MT IN MATH

of an algorithm as the first core element of AT. The second core element of AT we chose is the ability to specify a problem precisely, based upon which the algorithm can be constructed. This is also part of the definition of AT according to Futschek (2006) and we chose this ability as it is the most practical. It is the most practical, since this ability constitutes the link between (mathematical) problems people encounter on a daily base and AT. For the third core element of AT we chose the ability to construct an algorithm. This skill appears in all three definitions as mentioned before. To have the students reason at a higher level, we included for the fourth core element the ability to understand an algorithm. With understanding we mean the ability to improve the efficiency of given algorithms. We chose to include this aspect as the ability of making improvements requires reasoning at a higher level about algorithms, compared to reading or constructing algorithms. This element of AT is also present in the definitions of Lamagna (2015) and Futschek (2006). With that, these are the four core elements of AT we use to depict AT in this research from now on. So, to sum it up: the core aspects of algorithmic thinking we identified are:

AT1: the ability to read and execute an algorithm

The student is able to follow a short sequence of instructions)

AT2: the ability to specify a problem precisely

The student is able to break down a complex problem into component sub-problems and sub-tasks (decomposition) while separating the useful information from the not useful information (abstraction)

AT3: the ability to construct an algorithm

The student is able to construct an algorithm, using repetition/recursion if needed);

AT4: the ability to understand and improve an algorithm

The student is able to understand algorithms and therefore make improvements to the algorithm and/or generalize the algorithm).

Students will have to acquire the four above mentioned abilities to learn to think algorithmically. The level of difficulty of these skills differ, as the ability to read and to understand differ in level. Since the construction of an algorithm is not expected to be part of the prior knowledge of students, this skill is expected to be harder than reading an algorithm. To help students learn to construct algorithms, flowcharts will be introduced. Flowcharts serve as a helpful tool for creating algorithms, according to the teachers in the consortium. With that, Flowcharts

SUPPORTING STUDENTS' CT AND MT IN MATH

could be effective in terms of writing and understanding algorithms, as found in literature (Hall, 2007). Therefore, flowcharts will play a key role in our learning activities. As early as 1973, Furey, Wilson, Hillier, Kent, and Haussler (1973) found that a flowchart may be drawn to describe the program at a level of abstraction somewhere between the problem statement and the code of the completed program. The programmer designs the flowchart in such a way that it can be coded easily into a convenient programming language, yet keeps the underlying algorithm sufficiently transparent to think about in modular terms (Furey, Wilson, Hillier, Kent, & Haussler, 1973). Flowcharts are a form of unplugged programming; it does not rely on computational devices, whereas plugged programming does (Aranda & Ferguson, 2018). So, flowcharts may help to structure the algorithm through visualisation. For this reason, flowcharts will be included in the tasks. Enough exercises to let students practice with it should be included, as Farina (1970) explained that flowcharting is an art requiring practice.

So, both CT and MT include the ability to think algorithmically. For this study, we divided AT into four sub-skills, to make monitoring the development of AT among students more precise. By mastering the four subskills, we expect students to be able to think algorithmically. This leads us to our research question: *How can 16-17-year-old pre-university students acquire algorithmic thinking skills in the mathematics classroom?*

Methods

We performed an explorative design-based research, in which we collected qualitative data to answer the research question. The first phase of this research consisted of designing a teaching unit. This teaching unit consisted of activities focussing on the development of students' ability to think algorithmically. In the second phase of the research, the designed activities for the teaching unit were put into practice, while data were gathered through mini-interviews and written student work.

Teaching Unit Design

The teaching unit involved the approximation of the roots of a function by using the bisection method and the Newton-Raphson method. Task design was based on literature and on input and feedback from teachers from the consortium.

Literature suggests different design guidelines. For example, Voogt et al. (2017) suggest to make use of direct instruction for the introduction of new concepts, make use of a structure that

SUPPORTING STUDENTS' CT AND MT IN MATH

builds up from plain to more complex concepts, make use of exercises for the students to help them practice with the new concepts and make use of teaching methods in which students collaborate. According to Reppenning et al. 2010, materials supporting algorithmic thinking should make use of a low threshold, and a high ceiling. The consortium mentioned the use of flowcharts, exercises in which students have to compare results from ICT-tools or the results from an ICT-tool and a numerical outcome, the usage of existing algorithms and explaining them, programming to be optional and teaching methods in which students have to collaborate as important elements for task design.

A draft version of the materials was then tested and reflected upon on the LIO day; a yearly event on which math didactics and math teachers in training gather to keep up with the latest movements in math education. Feedback given here was used to refine the materials.

The teaching activities, as attached in Appendix A, made use of a structure that builds up from plain to more complex concepts. Chapter 1 starts with a general introduction to introduce algorithms to students by examples from their daily lives. Task 2 and 3, as can be seen in Appendix A, are examples of such tasks. An introduction to flowcharts follows in chapter 2. Task 5 and 6 from the teaching unit are examples of those tasks here. Chapter 3 explains existing mathematical algorithms. Tasks 9, 10 and 13 are examples of such tasks. More complex mathematical algorithms are put central in chapter 4 and 5. We chose the algorithms for the bisection method and the Newton-Raphson method respectively. Tasks here include the construction of the flowchart (tasks 15 and 17) and the construction of algorithms for those methods (tasks 14 and 18). In the final chapter, chapter 6, students are asked to compare the results from previous tasks to results from the TI graphing calculator. Task 22 is a good example of such a task here. Chapter 7 concerns programming using either Excel, Javascript, or the graphing calculator, and was optional.

Instruments

The main instruments are (1) a codebook for evaluating students' written work and (2) a mini-interview setup. Both instruments aimed at measuring the presence of algorithmic thinking.

As explained before, algorithmic thinking was divided into the four abilities AT1, AT2, AT3, and AT4, and the learning activities are calling upon either one or two of these abilities. The codebook, included in Appendix B, defines codes for the assessment of each of the four sub-abilities of algorithmic thinking. The codebook was used for analysing both the written student work as well as the answers given in mini-interviews regarding the tasks from the teaching

SUPPORTING STUDENTS' CT AND MT IN MATH

materials. A 0 was given to incorrect answers, answers that were solely partly convincing were coded with a 1 and convincing answers for the skills tested were coded with a 2.

Besides the students' answers on the tasks, we also wanted to gain insight into their thought processes. As Lye and Koh (2014) stated, to better examine students' performance, students could be asked to verbalize their thought process using think aloud protocol. For this reason, questions were prepared to ask students while they were doing particular tasks to reveal their thought processes. This method of data collection is called mini-interviews (Drijvers, 2003; Bakker, 2004). The mini-interviews were semi-structured, to leave space to anticipate on answers, and most of the questions of the interviews were open, as closed questions might limit the space for answers of the students. These mini-interviews were carried out as part of the observation, with random students. The questions asked during the mini-interviews were constructed in collaboration with my supervisors in advance and are included in Appendix C. Tasks in which students had to show understanding of algorithms were chosen to ask questions in mini-interviews about (tasks 2, 4, 10, 13, 14, 23 and 24). Topics here were the size of steps in step-by-step algorithms and students' reasoning about alterations to improve efficiency in algorithms. With that, students thought processes during the construction of the algorithms for the more complex mathematical contexts (the bisection method and the Newton-Raphson method) were questioned during the mini-interviews. Due to these questions, the mini-interviews may result in a learning effect, as is also shown in other research. Drijvers (2003) mentioned the danger that the mini-interviews in his research were guiding the students too much and Bakker and Gravemeijer (2004) acknowledged in their research that the mini-interviews probably had a learning effect. With that, the learning activities as a whole should contribute to a learning effect. However, as this research has an explorative character, possible learning effects from mini-interviews are not expected to harm the findings.

Participants

In collaboration with the schools of the teachers in the research consortium, 53 students in total, distributed over two groups from different schools, were designated as participants. The teaching unit was piloted on two 11th-grade classes of 33 students in total of a secondary school in a rural town in the west of the Netherlands. These two classes were similar as they had the same teacher and the same background in mathematics and programming; students had either math A or math B and all students had experience with programming in JavaScript, as they all followed the

SUPPORTING STUDENTS' CT AND MT IN MATH

course informatics. Therefore, results in these groups were expected to be similar. That is why these 33 students are from now on referred to as group 1.

The second group consisted of 20 students of a secondary school in a city in the west of the Netherlands and will hereafter be referred to as group 2. This group included both 9th and 10th graders. All students in this group followed the course math D, an optional math subject, usually taken by high achievers. The students in this group did not have prior knowledge with regards to working with the derivative, programming or using TI graphing calculators. Consequently, tasks in the teaching unit involving the derivative, programming and the use of TI graphing calculators were excluded for this group.

Group 1 was given the complete teaching unit, existing of 25 tasks, however, the teacher of this group included an alternative assignment for tasks 14 and beyond. This alternative assignment included creating flowcharts and codes for the bisection method and Newton-Raphson method, which is why this assignment was examined and then included in the data as the adjunct tasks from the teaching materials. For the second group, task 13 and task 17 until task 25 were excluded from the teaching materials, as these students did not possess prior knowledge with regards to advanced math and programming. So, the teaching unit consisted of 15 tasks for group 2.

Procedure

Prior to testing the teaching unit in their classes, the teachers concerned were asked to select from alternatives to carry out the teaching unit. Examples of such alternative are explaining the bisection or Newton-Raphson method with elaborated number examples or to make the programming chapter optional. These alternatives were given to let teachers have input in the execution of the materials in their classes. Teachers input here is valued as they are familiar with their students and can therefore make effective choices with regards to the prior knowledge and the condition of the groups.

In group 1, the students were put to work with different tasks in groups of two or three students, mixing the groups every ten minutes during the first lesson. For the second and third lesson, the students worked in set groups of two to three students. These teaching methods were chosen by the teacher. The students in group 1 were familiar with programming. For this group, an alternative final assignment was designed by the teacher, in which the students had to create a poster with the flowcharts and programs for the bisection and the Newton-Raphson method.

SUPPORTING STUDENTS' CT AND MT IN MATH

Students were put to work with this assignment, instead of the tasks of chapter 6 and 7 in which students had to do the same.

In group 2, the students were put to work in set groups of three to four students, during the entire teaching unit. As group 2 consisted of 9th and 10th grade students, the teacher of this group wanted each group to consist of at least one 10th grade student. This composition was chosen as the 10th grade students had learned how to work with the derivative yet, whereas the 9th grade students did not, which was necessary prior knowledge for some tasks. Furthermore, the students from group 2 did not have a TI Graphics Calculator yet and did not have prior knowledge with regards to programming. For this reason, some of the tasks were excluded for this group. The task about calculating the shortest distance between a given point and a line (task 13) was made optional, as the 9th grade students did not yet cover this subject in their math lessons. With that, the chapters about the Newton-Raphson method, the graphical calculator and programming (chapters 5, 6 and 7) were all excluded from the teaching unit as well. Lastly, for this group, the teacher added the component that the students would receive a grade for their work.

All lessons were observed by the first author. During the lessons in both groups, students' answers on the tasks were administered on paper and mini-interviews were carried out with random students both during and directly after the pilot. The questions in the mini-interviews were slightly altered after analysing the data of group 1 to make them more effective for group 2. An example of such an alteration is the shift from a closed to an open question, to gain more extensive information from the students.

Data

The main sources of data are (1) students' written work on the tasks and (2) the recorded mini-interviews. All the mini-interviews were audio recorded after permission was given from the students. Data was chosen to be collected in these two forms, as the use of data triangulation results in a more detailed description of the phenomenon studied. This detailed description enhances the validity of the research, as it increases the extent to which the study's findings are trustworthy and believable to others.

From a total of 33 students in group 1, 227 answers on tasks were collected and from a total of 20 students in group 3, 225 answers on the tasks were handed in. We mention this to give an indication of the work that has been done in each group. All the 452 students' written answers were then analysed according to the framework.

SUPPORTING STUDENTS' CT AND MT IN MATH

However, as the students from the first group did not receive a grade and were put to work in groups that changed every 10 minutes, a lot of answers to the tasks were not handed in. This caused quite some missing information. The collected data from the pilot in the first group were not complete, and therefore, not as usable as the written documents handed in by the second group.

In total, 42 mini-interviews were conducted, of which 34 aimed at specific topics in tasks and 8 were about students' opinions of the teaching unit. The 34 mini-interviews that aimed at specific topics were held during the pilot and the 8 mini-interviews regarding students' opinions on the teaching materials were held directly after the pilot. The amount of mini-interviews per group was not distributed evenly, as the first group consisted of more students. 36 mini-interviews were conducted in group 1 with two students per mini-interview and 6 mini-interviews were conducted in group 2 with 4 or 5 students per interview. The length of the mini-interviews varied from about forty seconds to nine minutes and these audio recordings have been transcribed. The distribution of these 34 mini-interviews over the tasks can be seen in Table 1, along with the topics.

Table 1. Distribution of mini-interviews over tasks

Task	# mini-interviews	Topic
Task 2	5	Could the efficiency of the given algorithm be improved?
Task 4	8	Do the different steps in an algorithm have to be of the same size?
Task 10	5	Did you recognize a pattern in the algorithm?
Task 13	1	Is it possible to make your algorithm more efficient?
Task 14	8	Did you recognize a pattern in the algorithm?
Task 23	2	Did you recognize a pattern in the algorithm?
Task 24	5	Is it possible to make your algorithm more efficient?

Less interview data is recorded on the topics 'shortest route algorithm and programming, due to limited time. All data collected in this research were stored onto BetaStor. Betastor is an external server, dedicated to the Faculty of Science of Utrecht University. Numbers were used to represent participants instead of names, to secure the ethical integrity of this research. For the first group, students were given codes from O.1 to O.33 and for the second group students were coded DH.1 to DH.20.

SUPPORTING STUDENTS' CT AND MT IN MATH

Data Analysis

After testing out the teaching unit, all answers on the tasks, in the form of written documents, were analysed according to the categories as described in the codebook. In this codebook, the tasks were categorized by the four core elements of AT, as is mentioned before. While designing the tasks for the teaching materials, we determined for each task which core element of AT was involved to be able to give the correct answer. In seven tasks, students had to read a given algorithm (tasks 2, 4, 8, 11, 12, 16 and 19). For this reason, these tasks were categorized under AT1. Examples of tasks here are answering questions about algorithms after reading them and combining algorithms with matching flowcharts. In twelve tasks, students had to specify a problem precisely (tasks 3, 6, 7, 9, 10, 13, 14, 15, 17, 18, 23 and 24). Here students were faced with a problem in words, by which they had to come up with a specification of the problem. Therefore, these tasks were specified as AT2. In 13 tasks, students had to construct algorithms by given situations (tasks 1, 3, 5, 6, 7, 9, 10, 13, 15, 17, 18, 23 and 24). So, these tasks were labelled with AT3. The fourth core element of AT, AT4, is present in 8 tasks (tasks 2, 4, 11, 12, 20, 21, 22 and 25). Examples of these tasks were improving given algorithms, which builds on understanding those algorithms. Some of the tasks were categorized in more than one category. Examples here were a given problem in words by which students had to construct an algorithm. Students first had to specify the problem, before constructing an algorithm. So, per subskill of AT, the written documents should make clear whether the concerned ability is present.

The analysis of the mini-interviews aimed at the thinking processes of students. The mini-interviews give a more qualitative and in-depth view on the thinking processes of students, complementing the results of the students' written work. For the analysis of the mini-interviews, we made use of a list of subjects from the teaching unit. These subjects are: Increasing efficiency, Steps in an algorithm, Pattern recognition, Shortest route algorithm, bisection method algorithm, Programming the bisection method algorithm and Programming the Newton-Raphson method algorithm (see Table 1). These subjects only refer to AT2, AT3, AT4 and not AT1, as AT1 is a lower level ability (reading an algorithm is easier than constructing or understanding one). On the basis of this list, we drawn up an inventory of given answers and established the core of what was told in the mini-interviews for each of these subjects.

To ensure the trustworthiness of the coding in this section, one of the supervisors acted as a second coder of the written documents. She was sent roughly 10% of the written documents of

SUPPORTING STUDENTS' CT AND MT IN MATH

the second group and coded these independently of the first author. These results were then compared to the codes from the first author and to measure the inter-rater reliability, the Cohen's kappa statistic was used. The value of this statistic was calculated to be 0,83, which can be translated to an almost perfect agreement between the coders, according to Landis and Koch (1977), therefore ensuring the trustworthiness of codes. We chose to only have the written documents coded by a second coder, as the mini-interviews often resulted in a learning effect, causing the outcomes here to be less objective already.

Results

Results from students' written work from group 2

As stated before, the written documents collected from the pilot in the first group were not as usable as the written documents handed in by the second group. Therefore, the results from written documents in this next section will be of the second group only. In the next section, the results from the pilot in de first group will be named briefly.

The results from the analysis of students' written work are displayed in Tables 2, 3, 4 and 5. Each table represents one subskill of AT, as categorized before, and each row represents a task from the teaching unit. The four numbers in each row show the number of absent answers, the number of tasks graded by a 0, the number of tasks graded by a 1 and the number of tasks graded by a 2, respectively. Not all tasks of the teaching unit are displayed in the tables, due to the exclusion of those for the second group. Since we do not have data on those tasks, they will not be displayed in the tables. The tables with results on all tasks per group are included in Appendix D. All tasks from the teaching unit in which students were supposed to develop the ability to read and execute algorithms (AT1) are shown in Table 2.

Table 2. Level of performance on AT1 per task for group 2

Task	AT1: READING AND EXECUTING			
	Absent answers	0	1	2
Task 2	0 (0%)	7 (35%)	8 (40%)	5 (25%)
Task 4	12 (60%)	0 (0%)	8 (40%)	0 (0%)
Task 8	2 (10%)	1 (5%)	0 (0%)	17 (85%)
Task 11	5 (25%)	1 (5%)	0 (0%)	14 (70%)
Task 12	10 (50%)	1 (5%)	4 (20%)	5 (25%)
Task 16	12 (60%)	1 (5%)	0 (0%)	7 (35%)

SUPPORTING STUDENTS' CT AND MT IN MATH

Noteworthy in Table 2 is that for task 4 no student received the full score of 2 points. This can be explained by the fact that students had to examine each other's algorithms in task 4, which none of them did thoroughly. The same happened in task 12, in which students had to execute each other's algorithms and make suggestions for alterations to improve the algorithms. After checking the algorithms students handed in, it became clear that alterations were possible, but not many students suggested any. This suggests that students did not read and/or execute each other's algorithms properly or students read and/or executed each other's algorithms incorrectly. Since the percentages are high for the other tasks, we can infer that students did not take the time to read and/or execute each other's algorithms in tasks 4 and 12. Therefore, we cannot say anything about AT1 based on the results in task 4 and 12. The limited amount of time for the pilot explains the high percentage of absent answers at task 16. The trend in percentages for the tasks scoring a 2, so, from task 2 to task 11, is increasing. With that, the amount of 0-scoring answers decreases as the pilot progresses. This seems to suggest an increase in the ability to read and execute an algorithm by the students over the course of the teaching unit. Therefore, results show that students learned to read algorithms from the teaching unit.

Table 3. Level of performance on AT2 per task for group 2

Task	AT2: SPECIFYING THE PROBLEM			
	Absent answers	0	1	2
Task 3	0 (0%)	1 (5%)	5 (25%)	14 (70%)
Task 6	1 (5%)	0 (0%)	4 (20%)	15 (75%)
Task 7	9 (45%)	2 (10%)	6 (30%)	3 (15%)
Task 9	2 (10%)	2 (10%)	7 (35%)	9 (45%)
Task 10	3 (15%)	0 (0%)	10 (50%)	7 (35%)
Task 13	19 (95%)	0 (0%)	1 (5%)	0 (0%)
Task 15	6 (30%)	0 (0%)	5 (25%)	9 (45%)

In Table 3, the results per task in which AT2 was tested are shown. From task 3 up until task 15, group 1 shows high percentages in answers scoring a 1 or a 2. However, the low percentages at task 13 stand out. The low percentages here are due to the fact that this task was made optional for the students of group 1, as they did not all learn about calculating the distance between a point and a line yet. The amount of answers scoring a 2 seems to decrease, which can

SUPPORTING STUDENTS' CT AND MT IN MATH

be explained by the fact that the difficulty of the tasks increased towards the end of the pilot. These high percentages from the beginning onward can be explained by the fact that specifying problems in mathematical context was not new to the students. For instance, some students made use of the ABC-formula or completing the square in solving those tasks, which they had already learned in math class. Therefore, results show that students can specify problems precisely, but not whether they learned it from the teaching unit.

Table 4. Level of performance on AT3 per task for group 2

Task	AT3: CONSTRUCTING			
	Absent answers	0	1	2
Task 1	0 (0%)	1 (5%)	8 (40%)	11 (55%)
Task 3	0 (0%)	3 (15%)	13 (65%)	4 (20%)
Task 5	3 (15%)	0 (0%)	8 (40%)	9 (45%)
Task 6	1 (5%)	9 (45%)	6 (30%)	4 (20%)
Task 7	9 (45%)	3 (15%)	4 (20%)	4 (20%)
Task 9	2 (10%)	3 (15%)	10 (50%)	5 (25%)
Task 10	3 (15%)	3 (15%)	8 (40%)	6 (30%)
Task 13	19 (95%)	0 (0%)	1 (5%)	0 (0%)
Task 14	3 (15%)	1 (5%)	6 (30%)	10 (50%)
Task 15	5 (25%)	0 (0%)	8 (40%)	7 (35%)

In Table 4, results from tasks in which students needed to construct algorithms in written form or via flowcharts are shown. The mistakes made in the answers scoring a 0 or a 1 were mostly fundamental errors in constructing algorithms. For example, students constructed flowcharts without arrows or algorithms with multiple end states. This might indicate poor instruction in the teaching unit. Since the amount answers scoring a 0 or a 1 slightly decreases and the amount of answers scoring a 2 slightly increases, we can say that there was a slight increase in the ability to construct algorithms among students. The difference in tasks over the course of the teaching unit is that they gradually become more complex in terms of context. Chapter 1 starts out with algorithms in contexts of students' daily lives. Later, the contexts make use of mathematical concepts, increasing in level of complexity. Since the tasks become more complex, the amount of answers scoring a 0 or a 1 slightly decreases and the amount of answers scoring a 2 seems to slightly increase, an increase in this skill seems to have occurred. Therefore, we can state that the presence

SUPPORTING STUDENTS' CT AND MT IN MATH

of this skill in tasks slightly increased during the teaching unit. So, we can deduce that some students learned to construct algorithms from the teaching unit.

Table 5. Level of performance on AT4 per task for group 2

Task	AT4: UNDERSTANDING			
	Absent answers	0	1	2
Task 2	0 (0%)	7 (35%)	8 (40%)	5 (25%)
Task 4	12 (60%)	3 (15%)	5 (25%)	0 (0%)
Task 11	8 (40%)	0 (0%)	10 (50%)	2 (10%)
Task 12	10 (50%)	3 (15%)	7 (35%)	0 (0%)

As for Table 5, it can be seen that not much answers were handed in at these tasks. In all these tasks, the understanding of algorithms was measured, and most students failed to perform these tasks in the right manner. This could have to do with the fact that understanding is a higher-level skill, compared to the other subskills of AT, and these tasks were too hard. Therefore, students' written answers here do not reveal a definite sign of students developing the skill to understand algorithms.

Results from students' written work from group 1

As stated before, written documents collected during the pilot held in the first group, were not complete, and therefore not as usable as the written documents collected in the second group. Therefore, this section will briefly summarize the data from the first group. In Table 6, all results from the written documents from group 1 are displayed.

Table 6. Results group 1

Task	AT1: READING AND EXECUTING			
	Absent answers	0	1	2
Task 2	10 (30%)	1 (3%)	18 (55%)	4 (12%)
Task 4	30 (91%)	0 (0%)	3 (9%)	0 (0%)
Task 8	15 (45%)	3 (9%)	0 (0%)	15 (15%)
Task 11	22 (67%)	0 (0%)	1 (3%)	10 (30%)
Task 12	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 16	31 (94%)	0 (0%)	2 (6%)	0 (0%)
Task 19	33 (100%)	0 (0%)	0 (0%)	0 (0%)

SUPPORTING STUDENTS' CT AND MT IN MATH

Task	AT2: SPECIFYING THE PROBLEM			
	Absent answers	0	1	2
Task 3	10 (30%)	0 (0%)	19 (58%)	4 (12%)
Task 6	13 (39%)	0 (0%)	17 (52%)	3 (9%)
Task 7	23 (70%)	1 (3%)	5 (15%)	4 (12%)
Task 9	17 (52%)	1 (3%)	9 (27%)	6 (18%)
Task 10	21 (64%)	1 (3%)	6 (18%)	5 (15%)
Task 13	22 (67%)	1 (3%)	7 (21%)	3 (9%)
Task 15	21 (64%)	0 (0%)	4 (12%)	8 (24%)
Task 17	28 (85%)	0 (0%)	5 (15%)	0 (0%)
Task 18	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 23	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 24	23 (70%)	0 (0%)	4 (12%)	1 (3%)
Task	AT3: CONSTRUCTING			
	Absent answers	0	1	2
Task 1	10 (30%)	0 (0%)	14 (42%)	9 (27%)
Task 3	10 (30%)	0 (0%)	19 (58%)	4 (12%)
Task 5	15 (45%)	0 (0%)	5 (15%)	13 (39%)
Task 6	13 (39%)	0 (0%)	17 (52%)	3 (9%)
Task 7	23 (70%)	1 (3%)	5 (15%)	4 (12%)
Task 9	17 (52%)	1 (3%)	9 (27%)	6 (18%)
Task 10	21 (64%)	1 (3%)	6 (18%)	5 (15%)
Task 13	22 (67%)	1 (3%)	7 (21%)	3 (9%)
Task 14	22 (67%)	0 (0%)	5 (15%)	6 (18%)
Task 15	21 (64%)	0 (0%)	4 (12%)	8 (24%)
Task 17	28 (85%)	0 (0%)	5 (15%)	0 (0%)
Task 18	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 23	32 (97%)	0 (0%)	0 (0%)	1 (3%)
Task 24	23 (70%)	0 (0%)	9 (27%)	1 (3%)
Task	AT4: UNDERSTANDING			
	Absent answers	0	1	2
Task 2	10 (30%)	1 (3%)	18 (55%)	4 (12%)
Task 4	30 (88%)	0 (0%)	3 (12%)	0 (0%)
Task 11	22 (67%)	0 (0%)	1 (3%)	10 (30%)
Task 12	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 20	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 21	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 22	33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 25	33 (100%)	0 (0%)	0 (0%)	0 (0%)

SUPPORTING STUDENTS' CT AND MT IN MATH

As can be seen from Table 6, there are many missing data on almost all tasks. This is explained by the fact that the students were not motivated enough to do and/or write down all tasks. Causing these data to be incomplete and therefore less usable. However, data collected in this group in the form of qualitative data was usable.

Results from qualitative data

In this section, the results from the mini-interviews during the pilot and directly after the pilot are discussed, starting with the results from the mini-interviews that were held during the pilot.

In Table 7, the results from the mini-interviews that were held during the pilot are shown. Each row in these tables represents the task about which the questions in the mini-interview were held. The answers given during the interviews were coded by our codebook as well, as the questions asked often regarded the tasks from the teaching materials. The three numbers in each row show the number of given answers graded by a 0, the amount of answers graded by a 1 and the amount of answers graded by a 2, respectively. As stated before, the mini-interviews may have resulted in a learning effect. This explains the high percentages of answers given in the mini-interviews graded by a 2, as can be seen in Table 7. Because of these learning effects, the differences between the two groups are negligible and that is why the results of all mini-interviews are combined, rather than split out for the two groups.

Table 7. Level of performance on AT1 per task in mini-interviews

	Task	Score		
		0	1	2
AT2: SPECIFYING	10	0 (0%)	0 (0%)	5 (100%)
AT3: CONSTRUCTING	13	0 (0%)	1 (100%)	0 (0%)
	14	0 (0%)	2 (25%)	6 (75%)
	23	0 (0%)	0 (0%)	2 (100%)
	24	0 (0%)	4 (80%)	1 (20%)
AT4: UNDERSTANDING	2	0 (0%)	1 (20%)	4 (80%)
	4	0 (0%)	0 (0%)	8 (100%)
	13	0 (0%)	1 (100%)	0 (0%)
	14	0 (0%)	2 (25%)	6 (75%)
	23	0 (0%)	0 (0%)	2 (100%)
	24	0 (0%)	4 (80%)	1 (20%)

SUPPORTING STUDENTS' CT AND MT IN MATH

Five mini-interviews were held regarding task 2, in which students were asked whether they could increase the efficiency of a given algorithm. Not all students thought that alterations to improve the efficiency of the algorithm were possible, which suggests lack of understanding. But these results could have been expected this early on in the teaching unit.

The eight mini-interviews on task 4 included the questions already given in task 4: is every step clear enough, are steps open for multiple interpretations, could your algorithm be improved by altering the steps, are the steps equal in size and is it necessary to have steps equal in size? The checking of each other's algorithms was executed very poorly. During the mini-interviews, the students unanimously claimed that all steps in self-constructed algorithms were clear, no self-constructed algorithm could be improved and no step was open for multiple interpretations. Answers on the question whether steps in a step-by-step procedure should be of equal size were sometimes incorrect in the beginning. But, eventually, all students came to the conclusion that steps differ in size and that it is not necessary to have all steps equal in size, as long as the steps are clear, which is mentioned before as the learning effect from the teaching activities used in this research. An example here is the concluding answer given by three students from the second group: "*De stappen hoeven niet even groot te zijn. Als de stap maar duidelijk genoeg is, maakt het niet uit dat ze niet even lang zijn.*" [The steps do not have to be of the same size. As long as the step is clear, it does not matter that they are not of the same length]. The fact that these students include the requirement that steps do not need to be of the same length, as long as they are clear, shows the presence of the ability to understand algorithms.

About task 10, five mini-interviews were conducted. The topic for task 10 was pattern recognition. Here all students claimed that they recognized math B in the tasks and that they needed math B to solve them. Examples here are by the students from the second group: "*Ja, gelijkstellen herkend van wiskunde B.*" [yes, we recognized the need to equate from math B]. And "*We gebruikten het standaardstappenplan van wiskunde B voor het uitrekenen van een snijpunt van grafieken.*" [We used the standard step-by-step procedures as learned in math B for the calculation of the intersection point of two graphs]. Both answers here indicate the presence of the ability to specify a problem, by using math.

After task 10, eight mini-interviews were held regarding task 14. in these mini-interviews students were asked to give a description of the constructed algorithms for task 14. This concerned an algorithm for finding the shortest route. Six of these algorithms were correct.

SUPPORTING STUDENTS' CT AND MT IN MATH

The seven mini-interviews regarding tasks 23 and 24 also asked students to describe their constructed algorithms. These algorithms regarded the bisection and the Newton-Raphson method. Here, only three algorithms were found to be correct, which can be explained by the level of complexity of these methods.

In addition to these mini-interviews that were held during the pilot, some interviews were held with students after the pilot, regarding their opinion of the teaching activities. During these interviews, students explained that they liked this new subject, but that the math was hard for students with math A. With these questions, students were also asked what they thought algorithmic thinking actually means. Some students were able to describe the core of algorithmic thinking. The student with code *O1.17* summarized algorithmic thinking as: “*Je moet gewoon heel stapsgewijs denken.*” [you just have to think step-by-step], and the student coded as *DH1.4* described AT as: “*Meer denken in stappen.*” [thinking more in steps]. These descriptions of algorithmic thinking come close to our definition and we can therefore infer that at least some students made the connection between algorithmic thinking and the teaching unit from the results from the mini-interviews.

Conclusion and Discussion

The research question we set out to address is: *How can 16-17-year-old pre-university students acquire algorithmic thinking skills in the mathematics classroom?* To investigate this, we defined AT as the ability to read, execute, construct, understand and improve algorithms, to specify a problem, and construct an algorithm based on that specification. This skill was divided into four subskills to make it more precise, and a teaching unit was designed to help students develop these four subskills. These subskills are: (a) the ability to read an algorithm, (b) the ability to specify a problem, (c) the ability to construct an algorithm, and (d) the ability to understand an algorithm. The teaching unit was tested in two groups, each from a different school, under different conditions (e.g. moments in time, length-of-time and chosen alternatives by the teachers).

From the written documents, we inferred that data suggest an increase in the ability to read and execute an algorithm by the students over the course of the teaching unit. Therefore, results show that students learned to read algorithms from the teaching unit. With that, results show that students can specify problems precisely, but not whether they learned it from the teaching unit. Also, we can state that the presence of the skill of constructing algorithms in tasks slightly increased during the teaching unit. Therefore, we can deduce that the students learned to construct algorithms from the teaching unit. Lastly, students' written answers did not reveal a definite sign of students developing the skill to understand algorithms.

From the mini-interviews, we inferred that students showed the ability to specify a problem, construct an algorithm and understand algorithms. Given answers during the mini-interviews showed the presence of the ability to understand algorithms and other answers indicated the presence of the ability to specify a problem, by using math.

Overall, the written documents of group 2 in combination with the results from the mini-interviews reveal a definite sign of the presence of the ability to think algorithmically, according to our codebook. And by looking closely to the course of the teaching unit, students seem to show an increase in three of the four different subskills of algorithmic thinking.

So, how can students acquire AT skills? This research shows that students can effectively acquire these skills within math class, i.e. within mathematical contexts. With that, the alternation between tasks based on unplugged and plugged seems to be a factor in acquiring AT skills. Easier contexts from students' daily lives to more complex mathematical contexts seem to have contributed to the development of AT skills as well, which we inferred from our results. Lastly, the

SUPPORTING STUDENTS' CT AND MT IN MATH

use of flowcharts was of importance for the effectiveness of the teaching unit. This was derived from literature and followed from the data in this research.

To summarize, the results from the study indicate the potential of the designed teaching unit to support students in developing algorithmic thinking skills in the mathematics classroom. With that, results imply a potential of the teaching unit in combination with the codebook as an instrument to check students' answers for the presence of algorithmic thinking.

Limitations

A limitation of this research is the amount of handed in tasks. Because the students in group 2 would receive a grade for the tasks, these students handed in written work on almost all tasks. The students from group 1 would not receive a grade and were therefore not as willing to complete all tasks, causing us to miss information about those tasks. Consequently, we assume that completing all tasks leads to a more effective teaching unit.

Another limitation of this research is the number of students we tested the teaching unit on. The number of students that were participants in our study, do not represent the entire target audience of our research, as the number of participants was too small. Moreover, the participants in this study were distributed among two groups from two different schools. As these groups differed in grade, and therefore in prior knowledge, the data we collected within each school were collected during different circumstances. Combining these results of both schools weakened our conclusion.

Future research

Results made clear that alterations on the teaching unit are required, so future research is needed. Most of the mistakes made by the students during the entire teaching unit concern fundamental errors with respect to flowcharts and algorithms. For example, students constructed flowcharts without arrows or algorithms with multiple end states. This might indicate insufficient instruction on the key elements of algorithms in the teaching unit, as stated before. From the mini-interviews we inferred that not all students grasped the essence of the tasks. This could be a signal of insufficient instruction in the teaching unit as well. Possible alterations for future research might therefore be to include more detailed instruction on learning how to use flowcharts and how to define an algorithm. Lastly, an interesting next step would be to make use of a pre- and posttest to investigate whether students learn to think algorithmically from the learning activities designed in this research.

SUPPORTING STUDENTS' CT AND MT IN MATH

References

- Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Educational Technology & Society*, 47-57.
- Aranda, G., & Ferguson, J. P. (2018). Unplugged Programming: The future of teaching computational thinking? *Pedagogika*, 279-292.
- Bakker, A. (2004). *Design research in statistics education: On symbolizing and computer tools*. Utrecht: CD-β-Press.
- Bakker, A., & Gravemeijer, K. P. (2004). Learning to reason about distribution. In The challenge of developing statistical literacy, reasoning and thinking. In D. Ben-Zvi, & J. Garfield, *The Challenge of Developing Statistical Literacy, Reasoning and Thinking* (pp. 147-168). Dordrecht: Springer.
- Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads*, 48-54.
- Black, J. A., & Champion, D. J. (1976). *Methods and issues in social research*. New York: Wiley.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 annual meeting of the American Educational Research Association*, (p. 25). Vancouver, Canada.
- Castillo, A. S., Berenguer, I. A., Sánchez, A. G., & Álvarez, T. R. (2017). Computational algorithmization: Limitations in problem solving skills in computational sciences majors at University of Oriente. *International Journal of Education and Development*, 166-184.
- Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). *Developing computational thinking in the classroom: a framework*. Southampton: Southampton Education School.
- Drijvers, P. H. (2003). *Learning algebra in a computer algebra environment: Design research on the understanding of the concept of parameter (Doctoral dissertation)*. Utrecht: CD-β Press.
- Drijvers, P. H. (2015a). Kernaspecten van wiskundig denken. *Euclides*, 4-8.
- Drijvers, P. H. (2015b). Denken over wiskunde, onderwijs en ICT. *Euclides*, 4-10.
- Dubinsky, E. (1991). Reflective abstraction in advanced mathematical thinking. In D. Tall, *Advanced mathematical thinking* (pp. 95-123). Dordrecht, Netherlands: Kluwer Academic Publishers.
- Eisenberg, M. (2002). Output devices, computation, and the future of mathematical crafts. *Int J Comput Math Learn*, 1-44.
- Furey, T. S., Wilson, R. K., Hillier, L. W., Kent, W. J., & Haussler, D. (1973). Flowchart techniques for structured programming. *ACM Sigplan Notices*, 12-26.
- Futschek, G. (2006). Algorithmic thinking: the key for understanding computer science. *International conference on informatics in secondary schools-evolution and perspectives* (pp. 159-168). Berlin, Heidelberg: Springer.

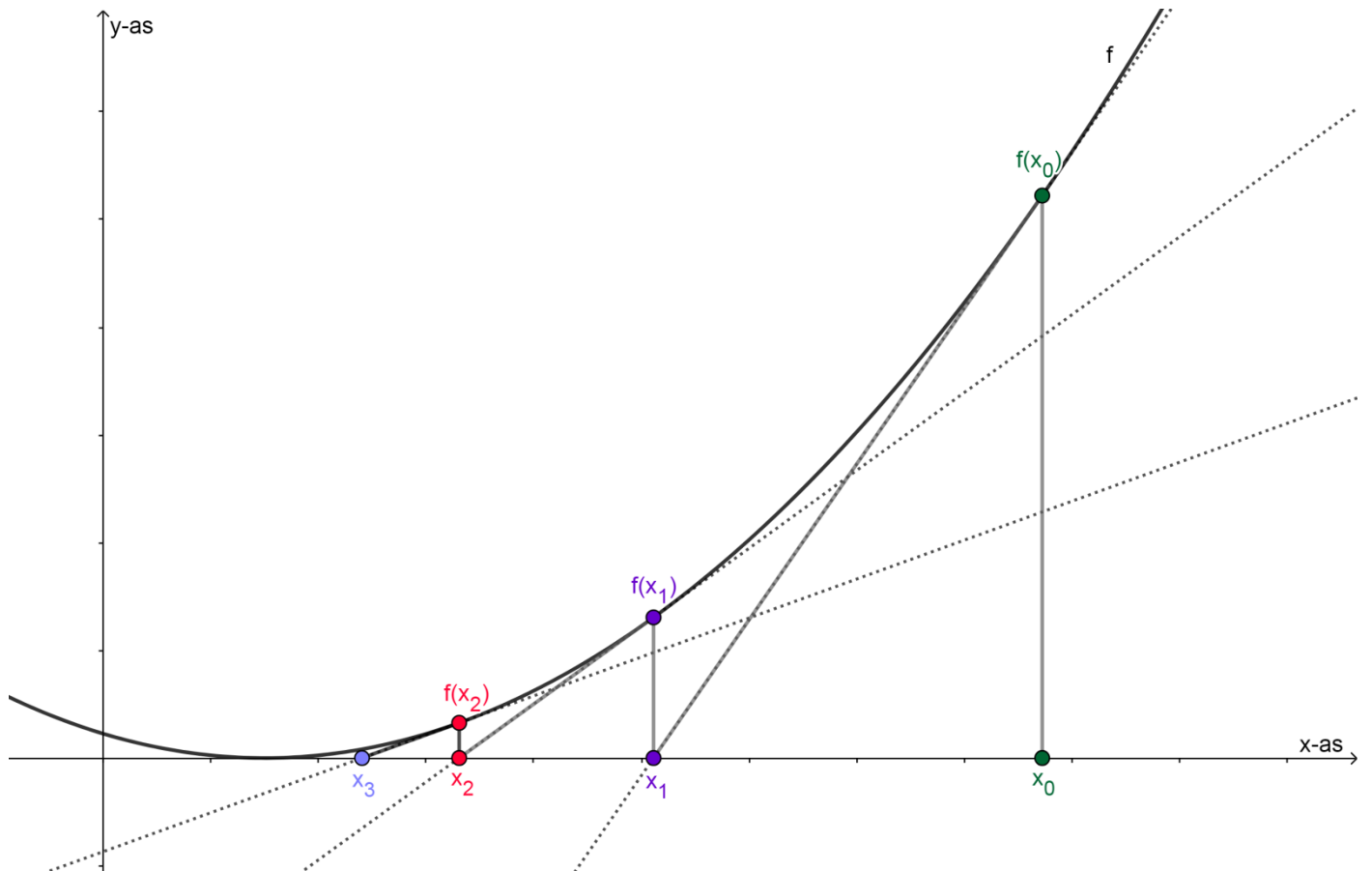
SUPPORTING STUDENTS' CT AND MT IN MATH

- Gray, E. M., & Tall, D. O. (1994). Duality, ambiguity, and flexibility: A "proceptual" view of simple arithmetic. *Journal for research in Mathematics Education*, 116-140.
- Hsu, C. C., & Wang, T. I. (2018). Applying game mechanics and student-generated questions to an online puzzle-based game learning system to promote algorithmic thinking skills. *Computers & Education*, 73-88.
- Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 583.
- Lamagna, E. A. (2015). Algorithmic thinking unplugged. *Journal of Computing Sciences in Colleges*, 45-52.
- Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 159-174.
- Lehner, P. N. (1979). *Handbook of Ethological Methods*. Garland: STPM Press.
- Lye, S. Y., & Koh, J. H. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 51-61.
- Mayfield, C., Hambrusch, S., Zhou, N., & Yadav, A. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, 1-16.
- Mulliner, E., & Tucker, M. (2017). Feedback on feedback practice: perceptions of students and academics. *Assessment & Evaluation in Higher Education*, 266-288.
- National Research Council. (1999). *Fluent With Information Technology*. Washington: National Academy Press.
- Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education*, 199-218.
- Orsmond, P. (2004). Evaluating our peers: is peer observation a meaningful process? *Studies in higher education*, 489-503.
- Pólya, G. (1963). On learning, teaching, and learning teaching. *The American mathematical monthly*, 605-619.
- Rahman, S. A. (2010). The development of a framework to assess mathematical thinking of stage one pupils. *Procedia-Social and Behavioral Sciences*, 307-311.
- Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. *the 41st ACM technical symposium on computer science education*, (pp. 265-269).
- Sadler, D. R. (2010). Beyond feedback: Developing student capability in complex appraisal. *Assessment & Evaluation in Higher Education*, 535-550.
- Sengupta P, K. J., Clark, D., Biswas, G., Kinnebrew, J. S., Basu, S., & Sengupta, P. (2013). Integrating computational thinking with K-12 science education using agent-based computation: a theoretical framework. *Journal of Information Technology Education*, 351-380.

SUPPORTING STUDENTS' CT AND MT IN MATH

- Sfard, A. (1991). On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educational studies in mathematics*, 1-36.
- Sherin, B. L. (2001). A comparison of programming languages and algebraic notation as expressive languages for physics. *International Journal of Mathematics*, 1–61.
- Stacey, K. (2006). *WHAT IS MATHEMATICAL THINKING AND WHY IS IT IMPORTANT?* Australia : University of Melbourne.
- Tall, D. (2013). *How humans learn to think mathematically: Exploring the three worlds of mathematics*. Cambridge: Cambridge University Press.
- Voogt, J., Brand-Gruwel, S., & Van Strien, J. (2017). *Effecten van programmeeronderwijs op computational thinking: een reviewstudie*. Den Haag: NRO.
- Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: learning biology through constructing and testing computational theories—an embodied modeling approach. *Cogn Instr*, 171–209.
- Wilensky, U., Brady, C., & Horn, M. (2014). Fostering computational literacy in science classrooms. *Communications of the ACM*, 17–21.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 33–35.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences*, 3717-3725.

Appendix A: Lessenserie Algoritmisch Denken



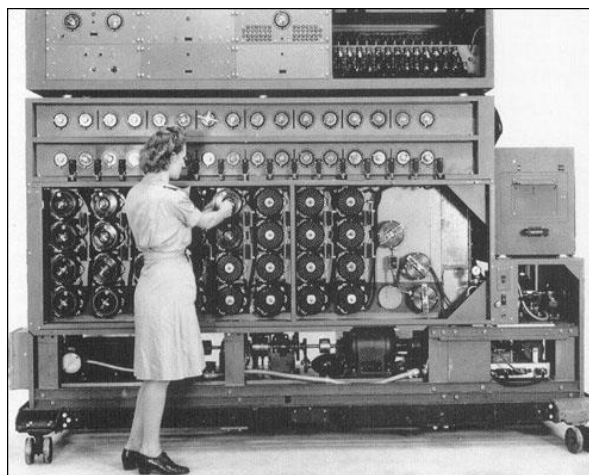
SUPPORTING STUDENTS' CT AND MT IN MATH

INHOUD

Introductie	30
1. Algoritmes	30
2. Stroomdiagrammen	32
3. Wiskundige algoritmes	33
4. De Halveringsmethode	35
5. Newton-Raphson methode	36
6. Grafische rekenmachine	37
7. Programmeren	37
Programmeren op je grafische rekenmachine	38
Programmeren in excel	39

Introductie

In onderstaand plaatje is één van de eerste computers afgebeeld, de Turingmachine. Alan Turing, een Britse wiskundige en computerwetenschapper uit 1900, is de bedenker hiervan en hij heeft tijdens zijn leven nog meer belangrijk werk verricht, hoewel dat leven niet altijd gemakkelijk is geweest. Turing is om drie redenen beroemd geworden. Ten eerste bedacht hij de Turing-test, een beroemd gedachte-experiment in de kunstmatige intelligentie. In dit experiment werd onderzocht of een computer menselijke intelligentie kan vertonen. Ten tweede was Turing in de Tweede Wereldoorlog betrokken bij het kraken van versleutelde berichten van de nazi's. De Duitsers codeerden hun berichten met de Enigma-systemen en Turing legde de basis voor het ontcijferen van deze berichten. Tot slot is hij beroemd geworden door het bedenken van de Turingmachine.



De Turingmachine is een eenvoudig apparaat en dus niet bedoeld voor praktische computertechnologie. Dit eenvoudige mechanisme zou in staat zijn om elk *algoritme* die kan worden beschreven uit te voeren. Je zou kunnen zeggen dat de Turingmachine algoritmes test. Je hebt misschien wel eens van een algoritme gehoord, maar wat betekent het nu precies?

1. Algoritmes

Een algoritme is een recept om stap voor stap tot een juist resultaat te komen. Een algoritme is een methode opgebouwd uit een vaste rij elementaire, opeenvolgende stappen die zeker tot de goede oplossing leidt. Zoals hiervoor is genoemd, kunnen algoritmes getest worden aan de hand van de Turingmachine, maar deze machine kan zelf geen algoritmes creëren. Waar een algoritme de beschrijving is van een recept tot het komen tot een juist resultaat, is een computerprogramma de implementatie van dat algoritme. Het vormgeven van algoritmes is dus een menselijke activiteit.

Er bestaan binnen rekenen-wiskunde talloze algoritmen, maar ook in het dagelijks leven komt men in aanraking met algoritmes. Voorbeelden zijn het volgen van een recept tijdens het koken of het volgen van een navigatiesysteem om een bestemming te bereiken. In beide gevallen doorloop je letterlijk stap voor stap een eindige reeks opeenvolgende instructies die vanuit een gegeven begintoestand naar een beoogd doel leiden. Een ander voorbeeld is de handleiding die je bij het kopen van bepaalde apparaten krijgt om je te helpen probleemsituaties om te zetten in oplossingen.

In het volgende voorbeeld wordt een algoritme beschreven waarin sprake is van herhaling, ook wel een *loop* genoemd.

SUPPORTING STUDENTS' CT AND MT IN MATH

→ **Voorbeeld 1.** Je moet je huiswerk maken. Je begint met opzoeken van het huiswerk op Magister of op je huiswerkplanner. Nadat je dit hebt opgezocht, pak je je boeken erbij en begin je aan de eerste huiswerktaak. Zodra je die af hebt, kijk je opnieuw op Magister of je huiswerkplanner naar de volgende huiswerktaak. Dit herhaal je tot er geen volgende opgave meer is en je ze allemaal hebt gemaakt. Je bent nu klaar met je huiswerk, dus je hebt je beoogde doel bereikt.

Het terugkijken op Magister of op de huiswerkplanner naar de volgende opdrachten van het huiswerk vormt in dit voorbeeld de herhaling. Zolang nog niet alles gemaakt is, ben je nog niet klaar met je huiswerk. Een dergelijk proces kan ook letterlijk in stappen beschreven worden. Zie **opdracht 1** hieronder.

Opdracht 1. Gebruik de vorm stap 1: ..., stap 2: ..., etc., om het proces van **voorbeeld 1** over het maken van huiswerk te beschrijven.

Opdracht 2. Kim heeft een doosje aardbeien gekocht en wil deze opeten. Voordat ze de aardbeien gaat opeten, moet Kim ze nog wassen en ontkronen. Ze heeft dit proces op de volgende manier beschreven;

Stap 1: Pak een aardbei uit het doosje

Stap 2: Was de aardbei

Stap 3: Ontkroon de aardbei

Stap 4: Leg de stukjes terug in het doosje

Stap 5: Controleer of er een aardbei in het doosje ligt die nog niet ontkroond is.

Stap 6: Zo ja, herhaal stap 2.

Stap 7: Zo nee, je bent klaar.

1. Klopt deze beschrijving volgens jou?

2. Kun jij het proces efficiënter beschrijven door een stap weg te nemen of aan te passen?

Opdracht 3. Vandaag heb je het eerste uur engels, het tweede uur wiskunde, het derde uur gym, het vierde uur een tussenuur en het vijfde uur biologie. Als je te laat bent voor de eerste les, moet je het zesde uur nablijven.

Beschrijf stap voor stap het proces dat nodig is voor het doorkomen van deze schooldag, waarbij je minimaal 5 stappen en maximaal 10 stappen gebruikt.






Opdracht 4. Om jouw algoritme voor de schooldag te testen leg je het aan een medeleerling voor. Laat hem/haar de stappen doornemen en beantwoord daarbij de volgende vragen.

- Is elke stap duidelijk genoeg?
- Bestaan er stappen in jouw algoritme die op een andere manier geïnterpreteerd kunnen worden dan hoe jij de stap bedoelt?
- Zouden er stappen toegevoegd of weggelaten kunnen worden waardoor jouw algoritme nog beter beschreven wordt?
- Zijn jouw stappen even groot?
- Is het nodig dat de stappen even groot zijn?

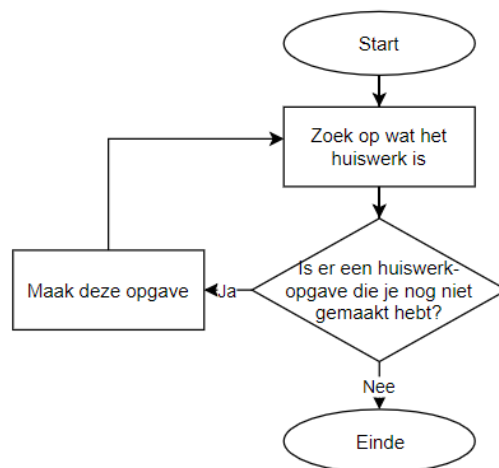
2. Stroomdiagrammen

In het voorgaande hoofdstuk hebben we algoritmes stap voor stap beschreven in tekst. Een algoritme kan ook schematischer worden weergegeven door een zogenaamd *stroomdiagram*.

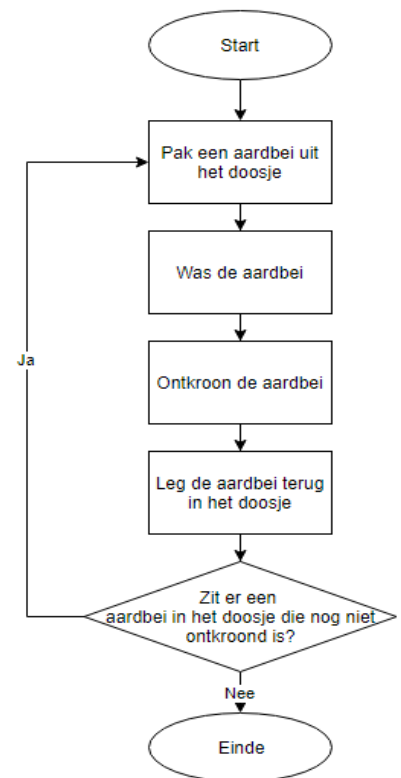
Een stroomdiagram of stroomschema is een schematische voorstelling van een proces. Ze worden in meerdere vakgebieden gebruikt om complexe processen overzichtelijk weer te geven. Hierdoor kunnen deze processen makkelijker bestudeerd en eventueel verbeterd worden. Stroomdiagrammen zijn een visuele voorstelling van handelingen en keuzes. Deze diagrammen maken gebruik van afgeronde rechthoeken, rechthoeken en ruiten om het soort stap aan te geven, en pijlen om de stroom en volgorde te bepalen. Zie de legenda hiernaast.

Symbool	Functie
	Begin/einde
	Handeling
	Keuze
	Invoer/uitvoer
	Richting

→ **Voorbeeld 2.** In **voorbeeld 1** werd een algoritme beschreven voor het maken van huiswerk. Dit algoritme kan ook worden weergegeven aan de hand van het stroomdiagram linksonder.



→ **Voorbeeld 3.** Een stroomdiagram dat past bij de context van **opdracht 2** over het wassen en ontkronen van aardbeien zie je hier rechtsboven.



Opdracht 5. Teken een stroomdiagram dat past bij jouw antwoord van **opdracht 3** over de schooldag.

Opdracht 6. Stel je voor dat je werkt bij een telefoonwinkel. Het is jouw taak om nieuwe verkopers het verkopen bij te brengen. Hiervoor maak je gebruik van een stroomdiagram. De nieuwe verkopers moeten dit diagram volgen om de klanten de juiste telefoon te kunnen verkopen, door het stellen van de juiste vragen. Voorbeelden van zulke vragen zijn: 'Welke telefoon heeft u nu?' en 'Heeft u een voorkeur voor een bepaald merk?'. Teken een stroomdiagram die past bij deze situatie.

Opdracht 7*. Beschrijf het algoritme voor het tekenen van een stroomdiagram.

3. Wiskundige algoritmes

Hoewel we tot nu toe alleen voorbeelden van algoritmes uit het dagelijks leven hebben gezien, kun je je misschien voorstellen dat er binnen de wiskunde ook veel algoritmes bestaan. We zullen eerst een blik werpen op mogelijke algoritmes voor het optellen van twee getallen. Daarna komen algoritmes voor complexere handelingen binnen de wiskunde aan de orde.

→ **Voorbeeld 4.** We bekijken het probleem $231 + 492$.

We lossen deze som op aan de hand van het volgende proces: zet beide getallen onder elkaar, zodat de eenheden onder de eenheden, de tientallen onder de tientallen enz. staan (1). De getallen die boven elkaar staan worden van rechts naar links bij elkaar opgeteld. De uitkomsten daarvan worden onder de streep genoteerd (2). Als één van de uitkomsten een 10 of groter is, dan wordt er 10 van dit getal afgetrokken en dit resultaat wordt onder de streep genoteerd (3). Vervolgens wordt er een 1 bovenaan de eerstvolgende linker kolom geschreven (4). Deze 1 tel je daarna bij dat rijtje op (5). Zie onderstaand proces.

(1)	(2)	(3)	(4)	(5)
$\begin{array}{r} 231 \\ 492 \\ \hline \end{array}$	$\begin{array}{r} 231 \\ 492 \\ \hline 3 \end{array}$	$\begin{array}{r} 231 \\ 492 \\ \hline 23 \end{array}$	$\begin{array}{r} 1 \\ 231 \\ 492 \\ \hline 23 \end{array}$	$\begin{array}{r} 1 \\ 231 \\ 492 \\ \hline 723 \end{array}$

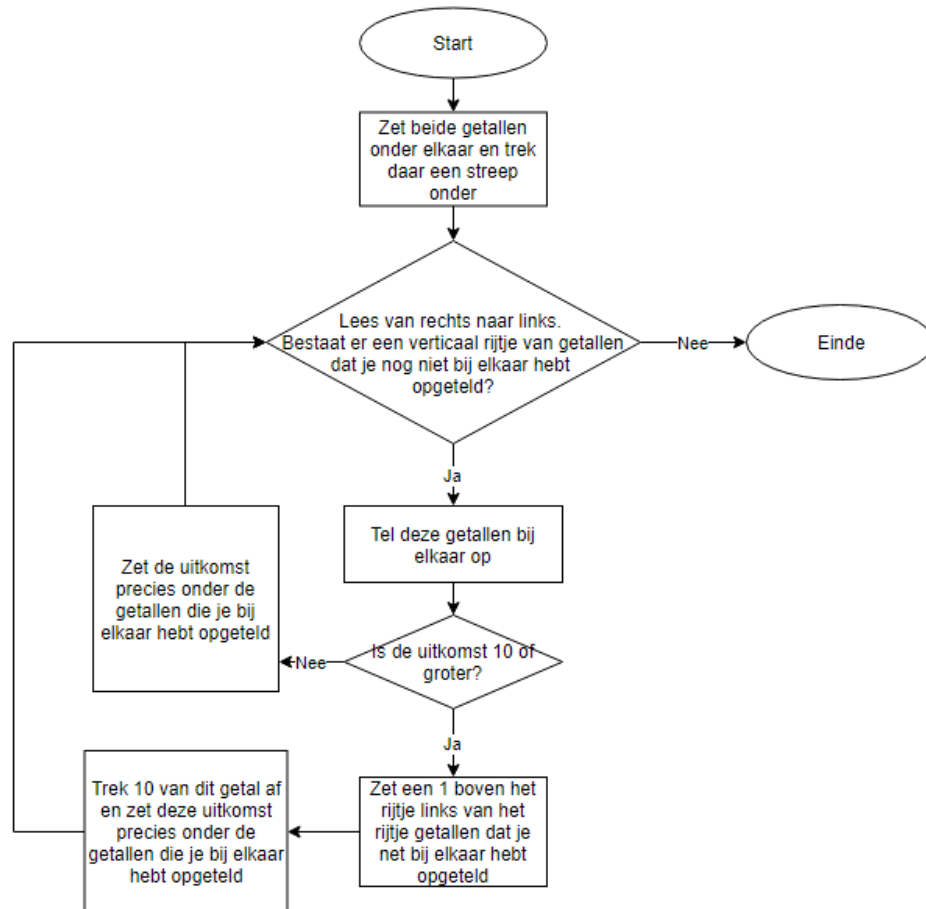
Een andere mogelijkheid begint opnieuw met het onder elkaar zetten van beide getallen, zodat de eenheden onder de eenheden, de tientallen onder de tientallen enz. staan (1). Nu veranderen we elke kolom van 2 getallen in een som, waarbij we rekening houden met welke getallen in eenheden, tientallen, honderdtallen, etc. gegeven zijn door het plaatsen van nullen (2). Deze sommen lossen we horizontaal op om de uitkomsten vervolgens verticaal bij elkaar op te tellen (3). Zie onderstaand proces.

(1)	(2)	(3)
$\begin{array}{r} 231 \\ 492 \\ \hline \end{array}$	$\begin{array}{r} 231 \\ 492 \\ \hline (1 + 2) \\ (30 + 90) \\ (200 + 400) \end{array}$	$\begin{array}{r} 231 \\ 492 \\ \hline 003 (1 + 2) \\ 120 (3 + 9) \\ 600 (2 + 4) \\ \hline 723 \end{array}$

Zoals je ziet wordt in beide gevallen $231 + 492$ juist berekend, maar op verschillende manieren. Ook deze wiskundige algoritmes kunnen we uitdrukken in stroomdiagrammen.

Opdracht 8. Bij welk van de twee bovenstaande algoritmes voor het oplossen van een som past onderstaande stroomdiagram?

SUPPORTING STUDENTS' CT AND MT IN MATH



Opdracht 9. Beschrijf het proces voor het vermenigvuldigen van twee getallen en teken er een stroomdiagram bij.

Opdracht 10. Leg in woorden uit hoe een algoritme eruit ziet voor het vinden van de coördinaten van een snijpunt van de grafieken van functies f en g met $f(x) = 2x^2 + 2x - 30$ en $g(x) = 14x + 2$.

Opdracht 11. Voer het algoritme dat je bij [opdracht 10](#) hebt opgesteld voor het vinden van de coördinaten van een snijpunt uit en verbeter het algoritme als dat nodig is.

Opdracht 12. Laat nu een medeleerling jouw algoritme van [opdracht 10](#) over het vinden van de coördinaten van een snijpunt uitvoeren. Is jouw algoritme duidelijk genoeg?

Bestaan er stappen in jouw proces die op een andere manier geïnterpreteerd kunnen worden dan hoe jij de stap bedoeld hebt?

Zouden er stappen toegevoegd of weggelaten kunnen worden waardoor jouw proces nog beter beschreven wordt?

Komt jouw medeleerling op het goede antwoord na het doorlopen van jouw stappen?

Zijn de stappen in jouw algoritme even groot? (kosten ze evenveel tijd, nadenkwerk, etc.)

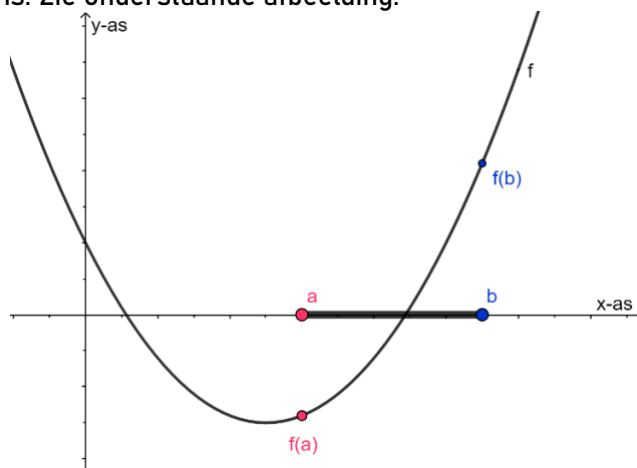
Is het nodig dat de stappen even groot zijn?

Opdracht 13. Ontwerp een mogelijk algoritme dat de kortste afstand berekent tussen lijn $l: y = -\frac{2}{3}x + 7$ en punt $A(5,8)$.

4. De Halveringsmethode

Nu we bekend geraakt zijn met algoritmes, kunnen we problemen met complexere algoritmes bekijken. Veel praktische problemen in de wiskunde kunnen worden herleid tot het zoeken van een of meer nulpunten van een functie. Daar zijn technieken voor ontwikkeld, die uiteenlopen van potlood en papier tot numerieke benaderingen door een computerprogramma. Voorbeelden van dergelijke algoritmes zijn de halveringsmethode en de methode van Newton-Raphson.

De halveringsmethode is gebaseerd op het feit dat de functiewaarden van een functie f negatief zijn aan de ene kant en positief aan de andere kant van een nulpunt. Na het vinden van twee functiewaarden die hieraan voldoen, wordt het interval gehalveerd door een grens te vervangen door het midden van het oorspronkelijke interval. Hierbij vervang je die grens waarvan je weet dat het nulpunt zich op het nieuwe interval moet bevinden. Hiermee ga je door tot een gewenste nauwkeurigheid bereikt is. Zie onderstaande afbeelding.



In andere woorden, als $f(a) < 0$ en $f(b) > 0$, of andersom, oftewel als $f(a) \cdot f(b) < 0$, dan heeft f een nulpunt op het interval $[a, b]$. Op deze gedachte is de halveringsmethode gebaseerd. Hierna bereken je het midden door $m = \frac{a+b}{2}$ en $f(m)$.

Er zijn voor $f(m)$ drie mogelijkheden:

1. $f(m) = 0$. In dit geval hebben we een nulpunt gevonden en zijn we klaar;
2. $f(a) \cdot f(m) < 0$. In dit geval bevindt zich een nulpunt op interval $[a, m]$. We vervangen b door m en we herhalen voorgaand proces op interval $[a, m]$;
3. $f(a) \cdot f(m) > 0$. In dit geval bevindt zich geen nulpunt op interval $[a, m]$. We vervangen a uit het oorspronkelijke interval door m en herhalen voorgaand proces met interval $[m, b]$, want daar moet dan wel een nulpunt zitten.

Deze methode stopt wanneer het interval $[a, b]$ heel klein geworden is.

Opdracht 14. Beschrijf in woorden een mogelijk algoritme voor de halveringsmethode.

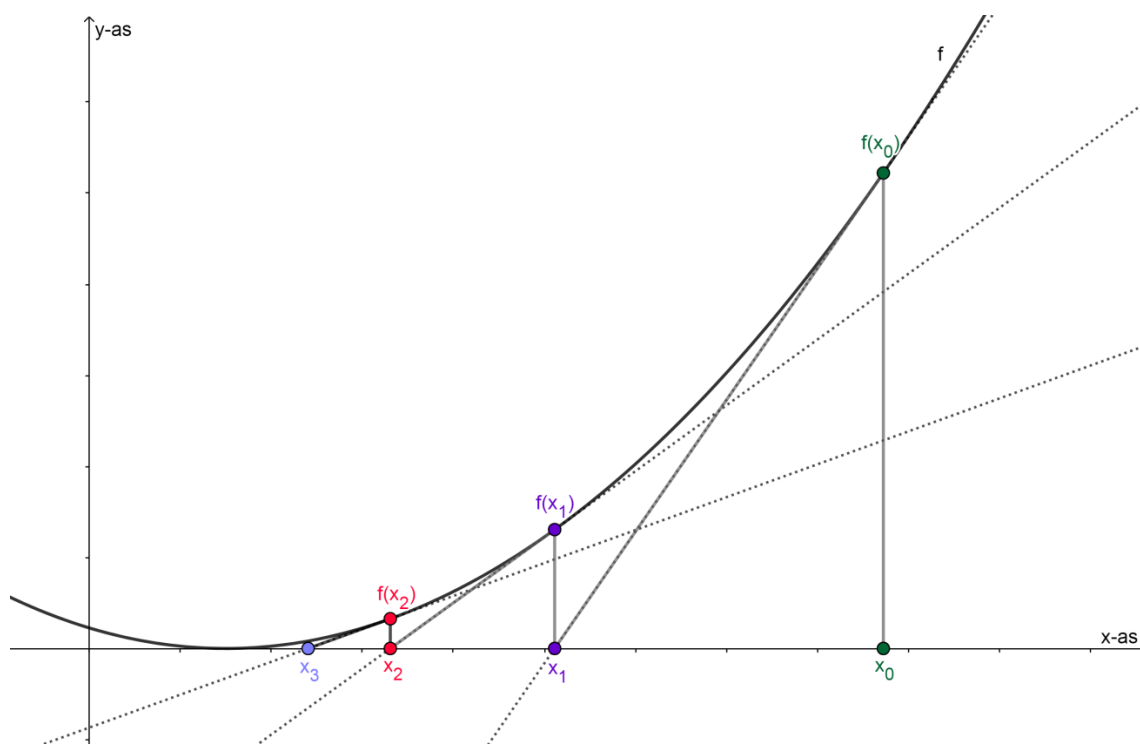
Opdracht 15. Maak een stroomdiagram dat past bij de halveringsmethode.

Opdracht 16. Pas jouw algoritme toe op de grafiek van $f(x) = x^3 - 3\frac{1}{2}x^2 + 4$ om het nulpunt op interval $[2,4]$ te bepalen, totdat je een interval hebt ter breedte van $\frac{1}{128}$.

NB. Een breedte van $\frac{1}{128}$ is kleiner dan $\frac{1}{100}$, dus deze benadering van het nulpunt ligt minder dan $\frac{1}{100}$ af van het werkelijke onbekende nulpunt. De benadering is daardoor nauwkeurig tot op (minstens) 1 decimaal.

5. Newton-Raphson methode

Naast de halveringsmethode zijn vele andere methodes ontworpen voor het vinden van nulpunten. Een hiervan is de Newton-Raphson methode. Deze methode is gebaseerd op het gebruik van raaklijnen. De snijpunten van raaklijnen met de x-as zouden het nulpunt steeds nauwkeuriger moeten bepalen, zoals te zien is in het figuur hieronder.



Opdracht 17. Maak een stroomdiagram dat bij past bij de Newton-Raphson methode.

Opdracht 18. Bedenk een algoritme voor het toepassen van de Newton-Raphson methode.

Opdracht 19. Pas jouw algoritme toe op de grafiek van $f(x) = x^3 - 3\frac{1}{2}x^2 + 4$ om de nulpunten te bepalen.

Niet alleen moet een algoritme een juist resultaat opleveren, het moet ook niet teveel moeite kosten. Een algoritme dat op een computer uitgevoerd wordt, moet binnen redelijke tijd met een resultaat komen. Bovendien is het wenselijk dat de hoeveelheid computergeheugen die daarvoor nodig is minimaal is. Je kunt je dus voorstellen dat meerdere algoritmes mogelijk zijn voor

SUPPORTING STUDENTS' CT AND MT IN MATH

hetzelfde proces om zeker tot een goede oplossing te leiden, maar dat ze niet allemaal even efficiënt hoeven zijn.

Opdracht 20. Vergelijk de eerste drie stappen van het toepassen van de halveringsmethode en de Newton-Raphson methode op de grafiek van $f(x) = x^3 - 3\frac{1}{2}x^2 + 4$, zoals je hebt gedaan in **Opdracht 16** en **Opdracht 19**. Welke methode vind je het beste en waarom?

6. Grafische rekenmachine

Hoe komt het dat een rekenmachine deze berekeningen dan wel weet te maken? Omdat er precies vaststaat wat er moet gebeuren volgens de algoritmes die in de rekenmachine zijn geprogrammeerd. Maar welk algoritme is geprogrammeerd en hoe dat algoritme exact werkt, weten maar weinig mensen.

In voorgaande hoofdstuk zijn verschillende manieren besproken waarmee de nulpunten van grafieken bepaald kunnen worden en daar gaan we in dit hoofdstuk mee verder. In dit hoofdstuk betrekken we de Grafische Rekenmachine bij het berekenen van nulpunten.

Opdracht 21. Gebruik je Grafische Rekenmachine om de nulpunten van de grafiek van $f(x) = x^3 - 3\frac{1}{2}x^2 + 4$ te berekenen. Wat valt je op als je deze uitkomst vergelijkt met die van de algoritmes uit voorgaand hoofdstuk?

Opdracht 22. Gebruik de halveringsmethode, de Newton-Raphson methode en je Grafische Rekenmachine om de nulpunten van de grafiek van $f(x) = x^3 - 3x + 2$ te bepalen. Vergelijk de uitkomsten met elkaar. Wat kun je hieruit opmerken?

7. Programmeren

Elk computerprogramma is eigenlijk één groot algoritme, maar dan omgezet in een programmeertaal. Voor de volgende opdrachten kies je uit het programmeren op je Grafische Rekenmachine of in Excel.

Opdracht 23. Programmeer de halveringsmethode op je Grafische Rekenmachine of in Excel.

Opdracht 24. Programmeer de Newton-Raphson methode op je Grafische Rekenmachine of in Excel.

Opdracht 25. Gebruik de optie CALC Zeroes op je Grafische Rekenmachine om de nulpunten van de grafiek van $f(x) = 4x^3 - 2x^2 + x - 12$ te bepalen. Vergelijk deze resultaten met elkaar. Van welk algoritme verwacht jij dat je Grafische Rekenmachine gebruik maakt om nulpunten te berekenen?

Programmeren op de grafische rekenmachine (TI)

Wat wil je bereiken?	Hoe bereik je dat?
Het programma begint met een leeg scherm	:ClrHome
Het programma maakt gebruik een beginwaarde. Het commando Prompt geeft gelegenheid aan het invoeren van meerdere beginwaarden.	:Input of :Prompt
Het programma slaat een waarde of tekst op om later te gebruiken of op terug te komen	→
Het programma weergeeft een tekst of een waarde	:Disp "tekst", of :Disp waarde
Het programma weergeeft een tekst of een waarde op een bepaalde plek, namelijk op regel x en teken y .	:Output(x,y ,tekst/waarde)
Het programma maakt gebruik van een If, een If,Then,End, of een If,Then,Else,End constructie. If geeft de voorwaarde aan	:If
Then geeft aan wat er gebeurt als aan de voorwaarde voldaan wordt	:Then
Else geeft aan wat er gebeurt als er niet aan de voorwaarde voldaan wordt	:Else
End geeft het einde van de constructie aan	:End
Het programma voert een instructie uit totdat aan een voorwaarde voldaan is. Na het commando Repeat volgt deze voorwaarde	:Repeat
Het programma voert een instructie uit zolang aan een voorwaarde wordt voldaan. Na het commando While volgt deze voorwaarde	:While
Het programma markeert een plaats om naar terug te kunnen springen	:Lbl
Het programma springt terug naar een aangegeven plaats in het programma, aangegeven met Lbl, om vanaf daar verder te gaan	:Goto
Het programma pauzeert tijdens de uitvoering tot de gebruiker op de enter toets drukt	:Pause

→ **Voorbeeld.** De ABC-Formule programmeren.

Het programmeren van de ABC-Formule in je Grafische Rekenmachine ziet er als volgt uit:

Commando	Betekenis
:ClrHome	Maak het beginscherm leeg
:Prompt A,B,C	A, B en C zijn de input voor dit programma
:(B ² -4AC)→D	Bereken de discriminant op basis van de input en sla deze waarde op onder de naam D
:Disp "D="	Weergeeft D= op het scherm
:Output(4,4,D)	Weergeeft de waarde van D op de 4de lijn en de 4de positie op het scherm
:If D>0	If maakt deel uit van de IF...THEN...ELSE-constructie en na If volgt de voorwaarde
:Then	Wat moet het programma doen als aan de voorwaarde voldaan is?
:Disp "OPLOSSINGEN:"	Weergeeft OPLOSSINGEN: op het scherm
:Disp $(-B+\sqrt{D})/(2A)$ ►Frac	Weergeeft eerste oplossing (in breukvorm) op het scherm
:Disp $(-B-\sqrt{D})/(2A)$ ►Frac	Weergeeft tweede oplossing (in breukvorm) op het scherm
:Else	Wat moet het programma doen als niet aan de voorwaarde voldaan is?
:Disp "GEEN REELE OPL."	Weergeeft GEEN REELE OPL. op het scherm
:End	Dit geeft het einde aan van de IF...THEN...ELSE-constructie

Programmeren in excel

- De halveringsmethode is gebaseerd op het feit dat als $f(a) \cdot f(b) < 0$ geldt, er zich een nulpunt

bevindt op het interval $[a, b]$. Als dit het geval is, wordt het midden van het interval $[a, b]$ gekozen als nieuwe grens voor het interval waarop het nulpunt zich bevindt. Hierbij wordt het nieuwe interval $[a, m]$ gekozen indien $f(a) \cdot f(m) < 0$ of $[m, b]$ indien $f(m) \cdot f(b) < 0$.

In Excel kan deze methode geïmplementeerd worden met behulp van de IF-functie. Deze functie begint met een voorwaarde, waarna het resultaat staat in het geval dat aan de voorwaarde wordt voldaan en eindigt met het resultaat in het geval dat niet aan de voorwaarde wordt voldaan. Na het koppelen van de juiste velden, kan Excel de halveringmethode toepassen voor het vinden van nulpunten. Een mogelijke lay-out hiervoor is de volgende:

a_0	b_0	$f(a_0)$	$f(b_0)$	m	$IF f(a_0) \cdot f(m) < 0, b_1, a_1$
:	:	:	:	:	:

- De Newton-Raphsonmethode is gebaseerd op het gebruik van raaklijnen. Hiervoor maakt deze

methode gebruik van de iteratie $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$. Na het koppelen van de juiste velden in Excel, kan Excel de Newton-Raphsonmethode toepassen voor het vinden van nulpunten. Een mogelijke lay-out hiervoor is de volgende:

x_n	x_{n+1}	$f(x_n)$	$f'(x_n)$
:	:	:	:

Appendix B: Codebook for analyzing students' written work

Task	Skill	No sign of skill	Answer shows skill not convincingly	Answer shows skill
1	AT3	<p>If the answer contains more than one of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer lacks a sign of AT3.</p>	<p>If the answer contains one of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows a sign of AT3.</p>	<p>If the answer does not contain any of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows AT3.</p>
2	AT1	<p>If the answer does not contain a comment similar to one of the following:</p> <ul style="list-style-type: none"> Stap 6: Zo ja, herhaal stap 2 klopt niet, or Vervang stap 6 en 7 door stap 6: Zo ja, ga terug naar stap 1 en zo nee, je bent klaar, or Vervang stap 6 en 7 door stap 6: Zo ja, herhaal stap 1 tot en met 5 en zo nee, je bent klaar <p>then the answer does not show AT1.</p>	<p>If the answer contains a comment similar to one of the following:</p> <ul style="list-style-type: none"> Stap 6: Zo ja, herhaal stap 2 klopt niet, or Vervang stap 6 en 7 door stap 6: Zo ja, ga terug naar stap 1 en zo nee, je bent klaar, or Vervang stap 6 en 7 door stap 6: Zo ja, herhaal stap 1 tot en met 5 en zo nee, je bent klaar <p>then the answer only shows a sign of AT1.</p>	<p>If the answer contains two or more comments similar to the following:</p> <ul style="list-style-type: none"> Stap 6: Zo ja, herhaal stap 2 klopt niet, or Vervang stap 6 en 7 door stap 6: Zo ja, ga terug naar stap 1 en zo nee, je bent klaar, or Vervang stap 6 en 7 door stap 6: Zo ja, herhaal stap 1 tot en met 5 en zo nee, je bent klaar <p>then the answer shows AT1.</p>
	AT4	<p>If the answer shows that the student thinks the given algorithm is correct and there is nothing to do to improve its efficiency, or the student suggests an improvement that is incorrect, then the answer shows no sign of AT4.</p>	<p>If the answer contains an alteration to one of the steps but the alteration could have been more efficient, then the answer only shows a sign of AT4.</p>	<p>If the answer contains an effective alteration to one of the steps, for instance:</p> <ul style="list-style-type: none"> Was alle aardbeien tegelijk in plaats van één voor één, or Het doel beschrijven bij stap 7: Zo nee, je bent klaar en kunt de aardbeien opeten,

				then the answer shows AT4.
3	AT2	If the answer does not show the situation to be broken down into component sub-problems and sub-tasks with the relevant information, then the answer shows no sign of AT2.	If the answer shows the situation to be broken down into component sub-problems and sub-tasks, but could have been done more efficiently (e.g. a compacter schedule), then the answer shows a sign of AT2.	If the answer shows the situation to be broken down into component sub-problems and sub-tasks with only the relevant information, then the answer shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
4	AT1	If the student was not able to answer the questions: <i>Is elke stap duidelijk genoeg?</i> and <i>Bestaan er stappen in jouw algoritme die op een andere manier geïnterpreteerd kunnen worden dan hoe jij de stap bedoelt?</i> , then the answer does not show the ability to read an algorithm.	If the student was able to answer some of the questions: <i>Is elke stap duidelijk genoeg?</i> and <i>Bestaan er stappen in jouw algoritme die op een andere manier geïnterpreteerd kunnen worden dan hoe jij de stap bedoelt?</i> correctly (yes and no answers are sufficient here), then the answer shows a sign of the ability to read an algorithm.	If the student was able to answer all of the questions: <i>Is elke stap duidelijk genoeg?</i> and <i>Bestaan er stappen in jouw algoritme die op een andere manier geïnterpreteerd kunnen worden dan hoe jij de stap bedoelt?</i> correctly, then the answer shows the ability to read an algorithm.
	AT4	If the student was not able to make any improvements on the algorithm with an incorrect or missing explanation, or the student answered the question: <i>'Is het nodig dat de stappen even groot zijn?'</i> with yes, then the answer shows no sign of AT4.	If the student was able to make an improvement on the algorithm, or can explain why not, or the student answered the question: <i>'Is het nodig dat de stappen even groot zijn?'</i> correctly, with a correct explanation, then the answer shows the presence of AT4.	If the student was able to make an improvement on the algorithm, or can explain why not, and the student answered the question: <i>'Is het nodig dat de stappen even groot zijn?'</i> correctly, with a correct explanation, then the answer shows AT4.

5	AT3	If the flowchart does not match the written algorithm for task 3, then the answer shows no sign of AT3.	If the flowchart matches the written algorithm for task 3, but contains small mistakes, then the answer only shows a sign AT3.	If the flowchart matches the written algorithm for task 3 and is correctly drawn, then the answer shows AT3. Note: If the written algorithm for task 3 was incorrect, but the flowchart here is correct (so the written algorithm for task 3 and the flowchart here do not match), then the answer also shows AT3.
6	AT2	The problem should be specified in three parts: welcoming the customer, helping the customer by asking questions and recommending the right phone for the customer. If the structure as described above is missing, the answer lacks a sign of AT2.	The problem should be specified in three parts: welcoming the customer, helping the customer by asking questions and recommending the right phone for the customer. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	The problem should be specified in three parts: welcoming the customer, helping the customer by asking questions and recommending the right phone for the customer. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
7	AT2	The problem should consist of drawing a start point, actions, arrows and an end point. If after drawing the arrows, an action is missing, there should be a possibility to add that later (by a loop), before drawing the end point.	The problem should consist of drawing a start point, actions, arrows and an end point. If after drawing the arrows, an action is missing, there should be a possibility to add that later (by a loop), before drawing the end point.	The problem should consist of drawing a start point, actions, arrows and an end point. If after drawing the arrows, an action is missing, there should be a possibility to add that later (by a loop), before drawing the end point.

		If the structure as described above is missing, the answer lacks a sign of AT2.	If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
8	AT1	The right answer here is that the flowchart matches the first written algorithm. If the answer is not correct, it does not show AT1.		The right answer here is that the flowchart matches the first written algorithm. If the answer is correct, it shows AT1.
9	AT2	The problem should consist of the input of the two numbers, multiplying each digit of the first number with each digit of the second number (variation is possible) and adding up the different outcomes. If the structure as described above is missing, the answer lacks a sign of AT2.	The problem should consist of the input of the two numbers, multiplying each digit of the first number with each digit of the second number (variation is possible) and adding up the different outcomes. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	The problem should consist of the input of the two numbers, multiplying each digit of the first number with each digit of the second number (variation is possible) and adding up the different outcomes. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), 	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), 	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts),

		<ul style="list-style-type: none"> the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer lacks a sign of AT3.</p>	<ul style="list-style-type: none"> the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows a sign of AT3.</p>	<ul style="list-style-type: none"> the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows AT3.</p>
10	AT2	<p>The problem should consist of the equation of both functions, which can be solved with the ABC-formula and results in the coordinates of the intersection point.</p> <p>If the structure as described above is missing, the answer lacks a sign of AT2.</p>	<p>The problem should consist of the equation of both functions, which can be solved with the ABC-formula and results in the coordinates of the intersection point.</p> <p>If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.</p>	<p>The problem should consist of the equation of both functions, which can be solved with the ABC-formula and results in the coordinates of the intersection point.</p> <p>If the structure as described above is there, the answers shows AT2.</p>
	AT3	<p>If the answer contains more than one of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer lacks a sign of AT3.</p>	<p>If the answer contains one of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows a sign of AT3.</p>	<p>If the answer does not contain any of the following mistakes:</p> <ul style="list-style-type: none"> there are not enough or clear subsequent steps in the algorithm, arrows are missing (only in context of flowcharts), the loop is missing while there should one be included, there is no or more than one start and final state, <p>then the answer shows AT3.</p>
11	AT1	<p>If the algorithm is not correctly executed, the answer shows no sign of AT1.</p>	<p>If the algorithm is correctly executed, but the outcome is not correct, the answer only shows a sign of AT1.</p>	<p>If the algorithm is correctly executed and the outcome is correct, the answer shows AT1.</p>
	AT4	<p>If no alterations were made to the algorithm, but there were possibilities to do so, the answer shows no sign of AT4.</p>	<p>If alterations have been made, but there are still possibilities to improve the algorithm in terms of efficiency, the answer only shows a sign of AT4. If no alterations were made, because no alterations</p>	<p>If alterations have been made, and there is no possibility left to improve the algorithm in terms of efficiency, the answer only shows a sign of AT4.</p>

			were possible, then the answer shows only a sign of AT4, if the student was able to explain this.	
12	AT1	If the algorithm is not correctly executed, the answer shows no sign of AT1.	If the algorithm is correctly executed, but the outcome is not correct, the answer only shows a sign of AT1. Note: if there is no execution of the algorithm visible, the answer is insufficient for this category.	If the algorithm is correctly executed and the outcome is correct, the answer shows AT1. Note: if there is no execution of the algorithm visible, the answer is insufficient for this category.
	AT4	If the student was not able to make any improvements on the algorithm, the answer shows no sign of AT4	If the student was able to make an improvement on the algorithm, it shows a sign of AT1 and AT4. If no alterations were made, because no alterations were possible, then the answer shows only a sign of AT4, if the student was able to explain this.	If the student was able to make multiple improvements on the algorithm, or can explain clearly why that is not possible, it shows AT4.
13	AT2	The problem should consist of finding the formula for the line tangent to the given line and situated through the given coordinates, the equation of that line to the given line, and using the Pythagorean formula to calculate the exact distance between those two points. If the structure as described above is missing, the answer lacks a sign of AT2.	The problem should consist of finding the formula for the line tangent to the given line and situated through the given coordinates, the equation of that line to the given line, and using the Pythagorean formula to calculate the exact distance between those two points. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	The problem should consist of finding the formula for the line tangent to the given line and situated through the given coordinates, the equation of that line to the given line, and using the Pythagorean formula to calculate the exact distance between those two points. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, 	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, 	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state,

		then the answer lacks a sign of AT3.	then the answer shows a sign of AT3.	then the answer shows AT3.
14	AT3	<p>If the answer contains more than one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer lacks a sign of AT3.</p>	<p>If the answer contains one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer shows a sign of AT3.</p>	<p>If the answer does not contain any of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer shows AT3.</p>
15	AT2	<p>The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough.</p> <p>If the structure as described above is missing, the answer lacks a sign of AT2.</p>	<p>The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough.</p> <p>If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.</p>	<p>The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough.</p> <p>If the structure as described above is there, the answers shows AT2.</p>
	AT3	<p>If the answer contains more than one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, 	<p>If the answer contains one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, 	<p>If the answer does not contain any of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state,

		then the answer lacks a sign of AT3.	then the answer shows a sign of AT3.	then the answer shows AT3.
16	AT1	If the answer does not contain the right coordinates (caused by misreading the algorithm) of the zero of the graph, the answer shows no sign of AT1. This question proceeds on the previous, this should therefore be taking into account while scoring this answer.		If the answer contains the right coordinates of the zero of the graph, the answer shows AT1. This question proceeds on the previous, this should therefore be taking into account while scoring this answer.
17	AT2	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is missing, the answer lacks a sign of AT2.	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
18	AT2	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of

		the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is missing, the answer lacks a sign of AT2.	the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	the zero to find the adjacent y-coordinate and finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
19	AT1	If the answer does not contain the right coordinates (caused by misreading the algorithm) of the zero of the graph, the answer shows no sign of AT1. This question proceeds on the previous, this should therefore be taking into account while scoring this answer.		If the answer contains the right coordinates of the zero of the graph, the answer shows AT1. This question proceeds on the previous, this should therefore be taking into account while scoring this answer.
20	AT4	If the answer contains no solid reasoning for the choice between the bisection and the Newton-Raphson method, the answers shows no sign of AT4.	If the answer contains a weak reasoning for the choice between the bisection and the Newton-Raphson method, the answers only shows a sign of AT4.	If the answer contains a solid reasoning for the choice between the bisection and the Newton-Raphson method from which the understanding of both methods appears, the answers shows AT4.
21	AT4	If the answer contains no solid reasoning for the similarities between the outcomes of the Newton-	If the answer contains a weak reasoning for the similarities between the outcomes of the Newton-	If the answer contains a solid reasoning for the similarities between the outcomes of the Newton-Raphson method and the TI Graphics Calculator

		Raphson method and the TI Graphics Calculator, the answers shows no sign of AT4.	Raphson method and the TI Graphics Calculator, the answers only shows a sign of AT4.	from which the understanding of both methods appears, the answers shows AT4.
22	AT4	If the answer contains no solid reasoning for the similarities between the outcomes of the bisection method, Newton-Raphson method and the TI Graphics Calculator, the answers shows no sign of AT4.	If the answer contains a weak reasoning for the similarities between the outcomes of the bisection method, Newton-Raphson method and the TI Graphics Calculator, the answers only shows a sign of AT4.	If the answer contains a solid reasoning for the similarities between the outcomes of the bisection method, Newton-Raphson method and the TI Graphics Calculator from which the understanding of both methods appears, the answers shows AT4.
23	AT2	The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough. If the structure as described above is missing, the answer lacks a sign of AT2.	The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.	The problem should consist of finding an interval $[a, b]$ such that $f(a) \cdot f(b) < 0$, then calculating the middle by $m = \frac{a+b}{2}$, then calculating $f(m)$ and replacing $f(m)$ by either $f(a)$ or $f(b)$ (depending on the sign) and repeating these steps until the new interval is small enough. If the structure as described above is there, the answers shows AT2.
	AT3	If the answer contains more than one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer lacks a sign of AT3.	If the answer contains one of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows a sign of AT3.	If the answer does not contain any of the following mistakes: <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, then the answer shows AT3.
24	AT2	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and	The problem should consist of a begin point on the graph, finding the zero of the tangent line to that point on the graph, using the x-coordinate of the zero to find the adjacent y-coordinate and

		<p>finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is missing, the answer lacks a sign of AT2.</p>	<p>finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, but not complete, or could have been improved in terms of efficiency, the answers shows a sign of AT2.</p>	<p>finding the zero of the tangent line to that point on the graph again, until the outcome is precise enough. If the structure as described above is there, the answers shows AT2.</p>
	AT3	<p>If the answer contains more than one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer lacks a sign of AT3.</p>	<p>If the answer contains one of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer shows a sign of AT3.</p>	<p>If the answer does not contain any of the following mistakes:</p> <ul style="list-style-type: none"> • there are not enough or clear subsequent steps in the algorithm, • arrows are missing (only in context of flowcharts), • the loop is missing while there should one be included, • there is no or more than one start and final state, <p>then the answer shows AT3.</p>
25	AT4	<p>If the answer contains no solid reasoning for which method the TI Graphics Calculator uses to calculate zeros, the answers shows no sign of AT4.</p>	<p>If the answer contains a weak reasoning for which method the TI Graphics Calculator uses to calculate zeros, the answers only shows a sign of AT4.</p>	<p>If the answer contains a solid reasoning for which method the TI Graphics Calculator uses to calculate zeros from which the understanding of both methods appears, the answers shows AT4.</p>

Appendix C: Mini-interview questions

Opgave Mini-interview vragen

- 2** Hoe komt het dat het proces efficiënter wordt beschreven door jouw aanpassing denk je? Zou er nog een andere aanpassing mogelijk zijn voor het efficiënter/in minder stappen beschrijven van het proces?
-
- 4** Wat is jullie antwoord op de vraag of stappen even groot moeten zijn? En waarom?
-
- 10** Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Kwam je tot een juist snijpunt door het uitvoeren van jouw algoritme? Waarom wel/niet? Hoe komt het dat jouw verbetering een verbetering voor dit algoritme vormt denk je?
-
- 13** Bestaan er stappen die weggelaten en/of toegevoegd kunnen worden om jouw algoritme nog beter te kunnen beschrijven? Hoe komt dat denk je?
-
- 14** Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Is het gelukt om met jouw algoritme een nulpunt te bepalen? Wat vind je van de stappen in jouw algoritme? Hadden dit er meer of minder kunnen zijn? Werkt jouw algoritme altijd denk je?
-
- 19** (18) Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Is het gelukt om met jouw algoritme een nulpunt te bepalen? Wat vind je van de stappen in jouw algoritme? Hadden dit er meer kunnen zijn of minder en werkt jouw algoritme altijd?
-
- 20** Welke methode (halverings- of Newton-Raphsonmethode) vind je het beste en waarom? Welke methode (halverings- of Newton-Raphsonmethode) vind je het beste en waarom als je kijkt naar beide algoritmes?
-
- 22** Wat kun je hieruit opmerken?
-
- (23)** Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Hoe ziet jouw algoritme eruit? Had dit efficiënter/in minder stappen gekund?
-
- (24)** Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Hoe ziet jouw algoritme eruit? Had dit efficiënter/in minder stappen gekund?
-
- (25)** Heb je gebruik gemaakt van een patroon of herken je een patroon in het algoritme? Van welk algoritme denk je dat je GR gebruik maakt om nulpunten te bepalen?
-

Appendix D: Written results on tasks

Table 8. Group 1 results on tasks

TASK	AT1: READING AND EXECUTIN				AT2: SPECIFYING				AT3: CONSTRUCTING				AT4: UNDERSTANDING			
	Absence	0	1	2	Absence	0	1	2	Absence	0	1	2	Absence	0	1	2
Task 1									10 (30%)	0 (0%)	14 (42%)	9 (27%)				
Task 2	10 (30%)	1 (3%)	18 (55%)	4 (12%)									10 (30%)	1 (3%)	18 (55%)	4 (12%)
Task 3					10 (30%)	0 (0%)	19 (58%)	4 (12%)	10 (30%)	0 (0%)	19 (58%)	4 (12%)				
Task 4	30 (91%)	0 (0%)	3 (9%)	0 (0%)									30 (91%)	0 (0%)	3 (9%)	0 (0%)
Task 5									15 (45%)	0 (0%)	5 (15%)	13 (39%)				
Task 6					13 (39%)	0 (0%)	17 (52%)	3 (9%)	13 (39%)	0 (0%)	17 (52%)	3 (9%)				
Task 7					23 (70%)	1 (3%)	5 (15%)	4 (12%)	23 (70%)	1 (3%)	5 (15%)	4 (12%)				
Task 8	15 (45%)	3 (9%)	0 (0%)	15 (45%)												
Task 9					17 (52%)	1 (3%)	9 (27%)	6 (18%)	17 (52%)	1 (3%)	9 (27%)	6 (18%)				
Task 10					21 (64%)	1 (3%)	6 (18%)	5 (15%)	21 (64%)	1 (3%)	6 (18%)	5 (15%)				
Task 11	22 (67%)	0 (0%)	1 (3%)	10 (30%)									22 (67%)	0 (0%)	1 (3%)	10 (30%)
Task 12	33 (100%)	0 (0%)	0 (0%)	0 (0%)									33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 13					22 (67%)	1 (3%)	7 (21%)	3 (9%)	22 (67%)	1 (3%)	7 (21%)	3 (9%)				
Task 14									22 (67%)	0 (0%)	5 (15%)	6 (18%)				
Task 15					21 (64%)	0 (0%)	4 (12%)	8 (24%)	21 (64%)	0 (0%)	4 (12%)	8 (24%)				
Task 16	31 (94%)	0 (0%)	2 (6%)	0 (0%)												
Task 17					28 (85%)	0 (0%)	5 (15%)	0 (0%)	28 (85%)	0 (0%)	5 (15%)	0 (0%)				
Task 18					33 (100%)	0 (0%)	0 (0%)	0 (0%)	33 (100%)	0 (0%)	0 (0%)	0 (0%)				
Task 19	33 (100%)	0 (0%)	0 (0%)	0 (0%)												
Task 20													33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 21													33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 22													33 (100%)	0 (0%)	0 (0%)	0 (0%)
Task 23					32 (97%)	0 (0%)	0 (0%)	1 (3%)	32 (97%)	0 (0%)	0 (0%)	1 (3%)				
Task 24					23 (70%)	0 (0%)	9 (27%)	1 (3%)	23 (70%)	0 (0%)	9 (27%)	1 (3%)				
Task 25													33 (100%)	0 (0%)	0 (0%)	0 (0%)

