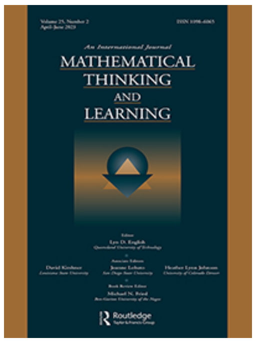


Mathematical Thinking and Learning

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/hmtl20

Teacher-student interaction supporting students' creative mathematical reasoning during problem solving using Scratch

Jan Olsson & Carina Granberg

To cite this article: Jan Olsson & Carina Granberg (2022): Teacher-student interaction supporting students' creative mathematical reasoning during problem solving using Scratch, Mathematical Thinking and Learning, DOI: 10.1080/10986065.2022.2105567

To link to this article: https://doi.org/10.1080/10986065.2022.2105567

MATHEMATICAL THINKING AND LEARNING
https://doi.org/10.1080/10986065.2022.2105567

RESEARCH ARTICLE

Teacher-student interaction supporting students' creative mathematical reasoning during problem solving using Scratch

Jan Olsson[a,b] and Carina Granberg[b,c]

[a]Division of Mathematics and Sciences with Didactics, Mälardalen University, Mälardalen, Sweden; [b]Umeå Mathematics Education Research Center (UMERC), Umeå University, Umeå, Sweden; [c]Department of Applied Educational Science, Umeå University, Umeå, Sweden

ABSTRACT

Studies have shown that learning mathematics through programming can be complex and that the programming itself might even hamper students' learning. However, few studies have focused on the role of the teacher and the teacher-student interaction that aims to support students' learning when using programming. The present study examines a didactic design in which a teacher uses well-prepared questions and suggestions that focus on students' reasoning to solve mathematical problems using Scratch, a block-based programming environment. Forty students, 10–11 years old, solved a geometry problem using Scratch supported by their teacher. The students' screen activities and teacher-student interactions were recorded. The results indicate that well-prepared general and task-specific questions targeting students' creative reasoning can help to overcome some of the complexities of learning mathematics while programming. The study furthermore shows that when students have sufficient programming skills, Scratch could even have the potential to support their reasoning and at the same time guide the teacher to give timely feedback.

ARTICLE HISTORY
Received 29 December 2021
Revised 20 July 2022
Accepted 20 July 2022

KEYWORDS
Programming; Scratch; mathematics education; Problem-solving; creative reasoning; Teacher-student interaction

Introduction

Programming as a means to engage students in problem-solving and develop mathematical understanding has become increasingly common in the elementary school curriculum during the last decades (Balanskat & Engelhardt, 2014). As programming becomes more common in teaching, interest in research that focuses on learning related to programming in mathematics education has increased (Moreno-León & Robles, 2016). There are studies showing that students who have used programming for problem-solving develop mathematical thinking (N. Calder, 2018), increase their understanding of mathematical concepts such as the position system (Benton et al., 2018) and concepts related to geometry and algebra (Zhong & Xia, 2018), and become better problem solvers in mathematics (Brown et al., 2008; Moreno-León & Robles, 2016). Moreover, programming is itself a problem-solving activity involving the construction of mathematical structures, for example, algorithms (Barr & Stephenson, 2011). So, a teaching design inviting students to solve mathematical problems using block-based programming, could allow students to practice their problem-solving skills and engage in mathematical reasoning while learning both programming and mathematics.

Teaching students to use programming for problem-solving is challenging, however. Brennan and Resnick (2012) described one of these challenges as a "tension between technology and learning" in the sense that a perception that programming is difficult could hamper the learning of mathematics.

CONTACT Jan Olsson jan.olsson@mdh.se Division of Mathematics and Sciences with Didactics, Mälardalen University, Mälardalen, Sweden; Umeå Mathematics Education Research Center (UMERC), Umeå university, Sweden

We confirm:the manuscript is not previously published or simultaneously submitted elsewhere, and all original research procedures were consistent with the principles of the research ethics published by the American Psychological Association

Teachers therefore need to choose problems that put reasonable demands on the programming skills the students need to develop (Leung, 2011). Previous studies have also shown that students who engage in programming do not necessarily discover the mathematical elements to be learned. It is therefore suggested that teachers be prepared to initiate and engage in fruitful teacher-student interactions to overcome these obstacles (Benton et al., 2018). Earlier studies that did not include programming (Olsson & Teledahl, 2018; Teledahl & Olsson, (2019).) investigated teacher-student interactions aiming to support students' reasoning during problem-solving. These studies point out the value of preparing teacher-student interactions that can support students' problem-solving process without providing them with a method to use and thereby depriving them of the opportunity to engage in problem-solving. Lithner (2008) presents a framework underpinning the idea that students learn better when they create their own methods than they do when the methods are provided to them. Lithner (Ibid.) describes this kind of reasoning as creative mathematical reasoning (CMR) and argues that, to support CMR, teachers need to consider how to support students' reasoning rather than explaining how to solve the task at hand. When adding technology such as programming, both programming and mathematics must be considered when preparing to support students' reasoning.

The present study includes 40 students, 10–11 years old, and focuses on the teacher-student interactions supporting them to engage in CMR during problem-solving using Scratch (a software used for block programming). The aim and research questions will be presented in more detail later.

Background

A wide range of research claims that students should engage in problem-solving rather than memorizing procedures to develop conceptual understanding (see, e.g., Boaler, 2002; Hiebert & Grouws, 2007; Schoenfeld, 1985 & Schoenfeld & Sloane, 2016). Furthermore, mathematical reasoning is pointed out as a key to developing mathematical understanding (Ball & Bass, 2003). Eliciting and supporting students' reasoning has been found challenging (Franke et al., 2009). Ball and Bass (2003) suggest that to understand students' mathematical reasoning teachers need content knowledge beyond pure mathematical knowledge, for example, why common errors occur and how to represent them in ways that students understand. Sacristán et al. (2009) suggest that interactive digital tools support teaching problem-solving and reasoning because they allow students to test their ideas and provide immediate feedback, which motivates students to understand and correct mistakes. Mathematics and programming are often considered to be related since both include problem-solving and because implementations of programming are often guided by mathematics (Misfeldt & Ejsing-Duun, 2015). In mathematics education there have been several attempts to include programming as means to learn mathematics through problem-solving (Moreno-León & Robles, 2016). As early as in the 1980s, research on students' programming using LOGO (a programming language developed for educational purposes) showed that students' learning of mathematics while programming benefited from their engagement in problem-solving and mathematical reasoning (S.Y. Lye & Koh, 2014; S. Y. Lye & Koh, 2018). However, LOGO was never widely used in mathematics teaching, perhaps because a teaching design appropriate to the use of LOGO had not yet been developed (Ibid.). Efforts in the 1990s to use script-based applications were questioned with arguments that the dual challenge of learning both programming and mathematics was hampering mathematics learning because learning to code is difficult (Resnick et al., 2009). More recently, visual-based applications for programming that are far easier to learn have renewed interest in incorporating programming into mathematics education (Weintrop & Wilensky, 2015).

Mathematical learning through Scratch(2015).

The graphical block-programming software Scratch has been proposed as a means to allow students to engage in mathematical reasoning (Erol & Çırak, 2021). The program is easy to learn and allows students to focus on being creative and to reason systematically (Kalelioglu & Gülbahar, 2014). In brief, Scratch allows students to create objects by putting together blocks that communicate with each other mathematically (Benton et al., 2018). The programming blocks can be manipulated to direct the way a sprite (e.g., a cat) moves across a computer screen and (if desired) draw lines and figures. For example, a student can choose a moving block that tells the sprite to move a number of steps and then add a turning block that makes the sprite turn a chosen number of degrees. Engaging students in programming may therefore entail opportunities to learn mathematics without being hampered by difficulties associated with coding.

There are different views on the kinds of mathematics students might learn from coding in Scratch. N. Calder (2018) observed that working with Scratch facilitated mathematical thinking for 10-year-old students. As with other dynamic software, the interactive process of coding and visual feedback in the Scratch environment engaged students in an iterative process of constructing a solution intertwined with emerging understanding (N. Calder, 2018). It has been argued that connecting different representations, such as code and visual output, promotes reasoning and understanding (Ainsworth et al., 1998; Benton et al., 2018). In N. Calder's (2018) the students often used a "guess, test, and modify" approach that sometimes moved the students toward generalizations. The students came up with mathematically based ideas on how the program would work, then they tested their ideas by creating and running the program in Scratch. Scratch provided immediate visual feedback telling the students how well their program worked, and based on that information the students could reflect on how to modify the program. Compared to the use of pen and paper, Scratch provides a more creative environment for the students to test and iteratively develop their ideas and their mathematical understanding of geometry, arithmetic, and coordinates. However, it was not clear whether students eventually learned mathematics or if they were using mathematics they already knew.

Even though it is unclear whether students learn mathematics from coding, several studies report that students deepen their knowledge of mathematics when coding in Scratch. Kong and Kwok (2022) found that students participating in a series of interventions using Scratch to learn the concept of primes significantly improved their knowledge. Students in the Scratch environment used a broader range of approaches to tasks compared to what is usually found in traditional teaching (Ibid.). Calao et al. (2015) found that students who solved problems aided by Scratch developed significantly better understanding of mathematical concepts and reasoning skills compared to a control group learning the same mathematical content not aided by Scratch. In addition, better problem-solving and modeling skills were noticed among the students in the intervention group (Ibid.).

Digital tools to teach mathematics

It has often been suggested that digital tools can facilitate inquiry-based teaching and problem-based learning (Martin & Grudziecki, 2006; Voogt & Pelgrum, 2005). If students do not need to engage in time-consuming pen-and-paper procedures they can more readily focus on higher-order goals (Drijvers, 2020). However, an early assumption now widely accepted is that a digital tool in itself is not sufficient to promote mathematics learning (Clark-Wilson et al., 2020). Mayer (2014)(2014). claims that properly designed digital tools promote active learning in interactive environments where students may control, manipulate, and receive immediate feedback from software. Such interactivity promotes learning through dialogs between students and between students and teachers (Moreno & Mayer, 2007(2007).). Digital tools intended to support interactive exploratory learning are often called dynamic software (e.g., GeoGebra) and are features for extensive research (Hillmayr et al., 2020). It has been found that dynamic software enhances learning and understanding of geometry, algebra, and calculus by providing direct feedback on manipulations (Buckley et al., 2004; Granberg, 2016) and that the approach to tasks enhances reasoning skills (Baghat & Chang, (2015).; Granberg & Olsson, 2015). Hillmayr et al. (2020) found in their review that the use of dynamic software had a greater effect on learning outcomes than drill and practice applications and tutorial systems. It seems that the suggested possibilities of dynamic software (e.g., enhancing reasoning skills and understanding) overlap some of the suggested advantages of using programming. For example, Kynigos and Grizioti (2018) suggest that coding in block applications engages students in similar dynamic manipulations (e.g., guess, test, and modify) as digital geometry software. They argue that similar benefits, such as identifying properties of geometrical objects and developing generalizations, can be achieved (Ibid.).

Teacher support to students working with digital tools

There has generally been greater interest in the digital tools and how students use them than in explaining the teacher's role (Clark-Wilson et al., 2020). For example, there is still a lack of knowledge about how teachers' practices are affected by digital aids, how new practices of teaching are developed, and how teachers adapt their existing teaching to new conditions (Clark-Wilson et al., 2014, 2020). When it comes to teaching programming to help students learn mathematics, only a few studies have looked more closely at the role of the teacher (Forsström & Kaufmann, 2018; Vatansever & Baltacı Göktalay, 2018). It is therefore of interest to investigate the teacher's role as well as how students approach mathematics when engaging in programming.

Interacting with software such as Scratch promotes dialogue and reflection (N. S. Calder, 2009). The teacher-student interactions can therefore take advantage of such software (Hoyles & Noss, 2003). For example, a teacher may ask questions encouraging students to explain their ideas about a specific part of the program, and the visible program blocks can support the students' explanations. N. Calder (2018) suggests that formative feedback encouraging students to explain their thinking often elicits interesting reasoning but states that students' answers are not always related to what is supposed to be learned. Sjöberg et al. (2018) investigated how students learned mathematics while programming in Scratch. They found that students often learned while collaborating and that they frequently referred to their coding when explaining their ideas to one another. However, the pedagogical implications were mainly general. For example, they found that students can help each other and teachers can make use of students' codes when interacting with the students. However, merely engaging students in programming is not enough for them to learn mathematics; on the contrary, the teachers need to be prepared to support students to express their mathematical thinking when programming (Benton et al., 2018). Benton et al. (2018) suggest an approach where the teachers carefully explore the relations between mathematics and programming activities so that they are prepared to support those relations through teaching techniques such as hugging techniques (e.g., connecting students' thinking to mathematical concepts) and bridging techniques (e.g., generalizing concepts). The teachers in their study did not prepare supporting techniques in detail but rather focused their preparation on identifying the relations between programming and mathematical thinking.

In summary, it seems that there is potential in teaching mathematics through problem-solving in combination with programming in Scratch, but there is a need for further investigation concerning the role of the teacher in such activities.

The present study focuses on a teaching design that utilizes the characteristics of mathematics and programming as problem-solving activities, and on teacher-student interactions that support students' mathematical reasoning during problem-solving using Scratch.

Framework

The present study is guided by the theoretical framework of creative mathematical reasoning (CMR), which argues that problem-solving that requires students to create (parts of) their methods is beneficial for learning (Lithner, 2008). To operationalize teaching for CMR, the present study adapts Simon's (1995) learning trajectories and suggests four principles to guide teacher-student interactions in ways that promote CMR. Since the present study concerns problem-solving and programming, Leung's (2011) epistemic modes to guide mathematical reasoning while interacting with technology are added to support the teaching and task design and to gain an understanding of how students can use Scratch for mathematical problem-solving.

Creative mathematical reasoning (CMR)

Lithner (2008, 2017) suggests that the task design will have an impact on students' reasoning and that the characteristics of their reasoning will affect their learning. Lithner (2008, 2017) describes how students' reasoning differs depending on whether or not a method for solving the task is provided. When students are provided with a method, they tend to engage in algorithmic reasoning (AR). That is, they use the provided procedure without attempting to understand the underlying mathematics. On the other hand, when students are given problems but no method to use, they need to engage in creative mathematical reasoning (CMR) in which they create (parts of) the method by themselves. Lithner (2008) describes creative mathematical reasoning (CMR) as fulfilling three criteria: (a) Creativity: the learner creates a reasoning sequence not experienced previously or re-creates a forgotten one; (b) Plausibility: there are predictive arguments supporting the choice of strategy and arguments for verification, explaining why the strategy implementation and conclusions are true or plausible; and (c) Anchoring: the arguments are anchored in the intrinsic mathematical properties of the components of the reasoning. This kind of reasoning, creative and anchored in mathematics, is more likely than AR to support students to develop mathematical understanding.

Although earlier studies have shown that students perform better on posttests if they have engaged in CMR (Jonsson et al., 2014; Norqvist et al., 2019; Olsson & Granberg, 2019) there are several questions to consider when implementing CMR in regular mathematics teaching. For example, earlier studies were experimental with less complexity compared to a classroom context, and in all studies, there was a group of students who failed during practice when attempting to solve problems requiring CMR (Jonsson et al., 2014; Olsson & Granberg, 2019). The latter results point to the need to support students' problem-solving by such means as implementing teacher-student interaction, peer support, and/or the use of aiding tools. Lithner (2008), however, defines reasoning as the line of thought and therefore does not include the verbalization of one's thoughts. But in order to interact with the teacher, students need to articulate their reasoning. Hence, to operationalize and meet the challenge of engaging in CMR, a framework for guiding teacher-student interaction to support students' CMR has been created in earlier studies (D'Arcy & Olsson, 2021); Olsson & Teledahl, 2018; Teledahl & Olsson, 2019) and further developed for the present study.

Reasoning in a hypothetical learning trajectory

We consider this part to be an adaptation of Simon's (1995) *hypothetical learning trajectory*. Simon suggests that when teachers have formulated learning goals and designed tasks to fit these goals they should anticipate a possible learning trajectory for their students. Such preparation would help teachers identify students' difficulties and adjust their support accordingly. Our adaptation of Simon's model suggests that students' learning is dependent on how they engage in creative mathematical reasoning. Hence, we aim to anticipate students' *hypothetical reasoning trajectory* by identifying reasoning activities that are necessary to engage in but that some students will find challenging. Thereafter, we aim to design questions and feedback intended to initiate a teacher-student interaction that will help students who are stuck to resume their problem-solving. Sacristán et al. (2009) propose that establishing a hypothetical learning trajectory for tasks to be solved with the aid of digital tools will help the teacher recognize and support students who deviate from the learning trajectory. Furthermore, digital tools may add opportunities for students to develop mathematical reasoning, and teachers can take advantage of such opportunities by timely offering appropriate prompts and suggestions (Sacristán et al., 2009).

We suggest that our contribution to Simon's model could be described as: Explicit learning goals and tasks suitable to reach these goals by engaging in CMR (see *Method*), a learning process expressed as creative reasoning trajectories, and prepared teacher-student interactions in the form of questions and feedback to support reasoning trajectories (see, Table 1). The identified situations, as parts of creative reasoning trajectories, are formulated as four principles. These principles are used to guide the teacher-student interactions in the present study and are presented below.

Principles to guide teacher-student interaction supporting CMR

The operationalization of CMR in classroom teaching, represented by the four principles, has been developed in several iterations (D'Arcy & Olsson, 2021); Olsson & Teledahl, 2018; Teledahl & Olsson, 2019). The main overarching principle is that in order to learn mathematics, *students need to create and express independent reasoning*. That process is guided by four principles: *initiating* reasoning, *developing* reasoning, *verifying* reasoning, and *justifying* reasoning.

The principle of initiating reasoning

The overarching principle implies that students need to create and express independent reasoning. To initiate that reasoning, the students need to understand task-specific mathematical concepts and what the problem is asking for. Having that understanding presupposes that the students have and can recall specific mathematical prior knowledge.

The teacher-student interaction supporting this principle is intended to bring the students to share, as specifically as possible, what they have thought and might have done so far. The teacher initiates the interaction by asking prepared general and, if necessary, task-specific questions (see, Table 1 for examples). The questions are intended to support students as they process their understanding of the problem, to activate useful prior knowledge, and to provide the teacher with information about the students' possible lack of knowledge and need for clarification or instructions.

However, even when students manage to initiate their reasoning, their reasoning may be fragmentary, incorrect, or insufficient and in need of further development.

The principle of developing reasoning

When students' reasoning is insufficient to solve the problem, they need to *develop their reasoning*. The teacher-student interaction therefore aims to make sure that the students have understood the problem correctly and that their reasoning is anchored in the mathematics of the problem, and it aims to encourage the students to continue their problem-solving. The teacher initiates the interaction by asking prepared general and, if necessary, task-specific questions targeting the mathematics as well as the programming of the given problem. (See, Table 1 for examples.)

The principle of verifying reasoning

When students are exploring their ideas about how to solve the problem and/or they have solved (parts of) the problem but are not sure whether it is correct, they need to *verify their reasoning*. When the students have not tried to verify their ideas by themselves, the teacher-student interaction should include questions to encourage them to find ways to test and verify their reasoning. (See, Table 1 for examples.)

The principle of justifying reasoning

When students arrive at a solution and think that their answer is correct, they usually want some confirmation, either from the textbook answers or from the teacher. In order to engage in CMR, however, the students need to *justify their reasoning*. That is, they need to support their solution by presenting arguments anchored in the mathematics underlying the problem at hand. The aim of the teacher-student interaction is therefore not merely to provide confirmation but to ask questions that challenge students to argue that their solution works and to explain why. (See, Table 1 for examples.)

Table 1. The four principles of reasoning and examples of general and task-specific questions and suggestions aiming to support each reasoning principle.

Reasoning principle	General questions/suggestions	Task-specific questions/suggestions
Initiating reasoning:	Have you read the task?	What do you know about squares?
	What are you supposed to do?	How do you want the cat to move?
	Read the instructions out loud.	Do you know how to make the cat move?
	Tell me what you have thought so far.	What do you need to do to make a (line, turn, square, rectangle, triangle)?
Developing reasoning:	Tell me what you have thought so far.	How do the sides of a square relate to each other?
	Tell me what you have done so far.	What should the corners (angles) of a square look like?
	Tell me how you reasoned when you constructed your program.	If you double all sides, what happens with the area? Perimeter?
Verifying reasoning:	Have you tried to test your idea/solution?	Can you use the cat (create a/adjust your program) to test your idea?
	Can you find a way to test your idea/solution?Are you sure your solution is correct?	
Justifying reasoning:	Are you sure your solution is correct?	Tell me, why is this figure a square?
	Tell me why your solution is correct/not correct.	Convince me that this figure is twice the size of the other.

The principles guiding the teacher-student interaction

When the student encounters any difficulty, the teacher needs to gather information about the student’s reasoning to be able to choose which principle is appropriate to guide the teacher-student interaction. Once the student has initiated her reasoning, the principles have no mutual hierarchy. For example, if the student has ideas on how to solve (parts of) the problem but is not sure that it is correct, then the teacher encourages her to find a way to *verify* her reasoning. Thereafter the student might use the outcome to *develop* her reasoning. If the student needs to *develop* her reasoning, then the teacher might address her understanding of the given problem, her prior knowledge, and when the student comes up with an idea (correct or incorrect) the teacher will suggest she find a way to *verify* her reasoning. If the student believes that (parts of) her solution is correct, the teacher challenges her to *justify* why the solution is correct. Then it is possible that the student has gained insights that she can use to *develop* her reasoning further. Figure 1 visualizes the way the principles interact.

Epistemic modes characterizing acquisition of mathematics when using digital tools

Earlier research has pointed out the challenge of initiating mathematics learning when students use digital aids (Barr & Stephenson, 2011; Brennan & Resnick, 2012). Students need to sufficiently master the technology to be able to benefit from it. Leung (2011) suggested three epistemic modes which he claims “characterize [the] mathematics knowledge acquisition process” (Ibid., p. 327) when it comes to techno-pedagogic task design.

The modes are: the establishing practices mode (EPM), the critical discernment mode (CDM), and the situated discourse mode (SDM). In the *establishing practices mode*, the students engage in the construction and manipulation of mathematical objects and in doing so gain routines to interact with the technology. In the present study, this mode will concern students’ familiarization with Scratch as they choose and combine blocks to create a program that will make a cat move across the screen. In the *critical discernment mode*, the students move their focus from handling the technology to identifying similarities, patterns, and variations among the constructed objects and to identifying information needed to solve the problem. Finally, in the *situated discourse mode*, the students use gained information to develop reasoning that involves

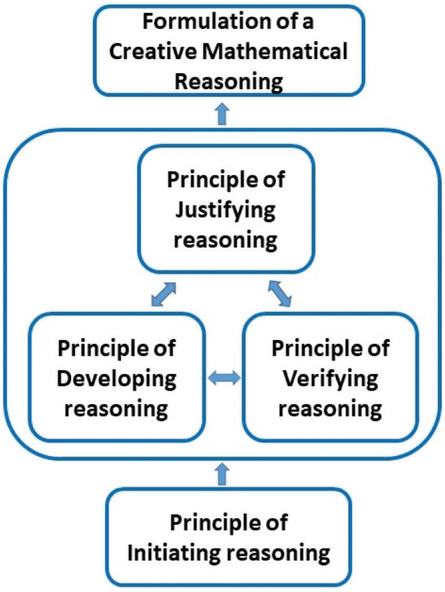

Formulation of a Creative Mathematical Reasoning
Principle of Justifying reasoning
Principle of Developing reasoning
Principle of Verifying reasoning
Principle of Initiating reasoning

Figure 1. The four principles, initiating, developing, verifying, and justifying reasoning, aiming to bring the students to create and formulate a creative mathematical reasoning. Once the student initiates her reasoning the other principles have no mutual hierarchy.

generalizing assumptions. For example, students in this study concluded that they always need 90 degrees to make the sprite make a “straight” turn, that the corners of squares as well as rectangles are always 90 degrees, etc. Students also use the information to adjust constructions with the purpose of supporting generalizations. Leung suggests that transitions between the modes entail cognitive effort, which can be supported by the teacher’s encouraging feedback and requests for justifications (Leung, 2011).

AIM and research questions

The aim of the present study was to develop our understanding of whether and how teacher-student interaction aimed at encouraging students’ CMR will support students’ problem-solving using Scratch. A further aim was to gain knowledge about whether and how students, with and without this kind of teacher-student interaction, will use the mathematics their programming represents to develop, justify, and verify their reasoning.

RQ 1: How can teacher-student interaction support students to initiate and develop their creative mathematical reasoning?

RQ 2: How can teacher-student interaction support students to justify and verify their creative mathematical reasoning?

RQ 3: How do students make use of Scratch to engage in creative mathematical reasoning?

Method

The intervention lesson in focus for the present study was preceded by a programming lesson to familiarize the students with Scratch and followed up by a consolidation lesson based on insights gained during the programming. These insights were used to introduce concepts such as angles and areas. The intervention is presented below, including the design of tasks and teacher-student interaction, the method of data collection, and data analysis.

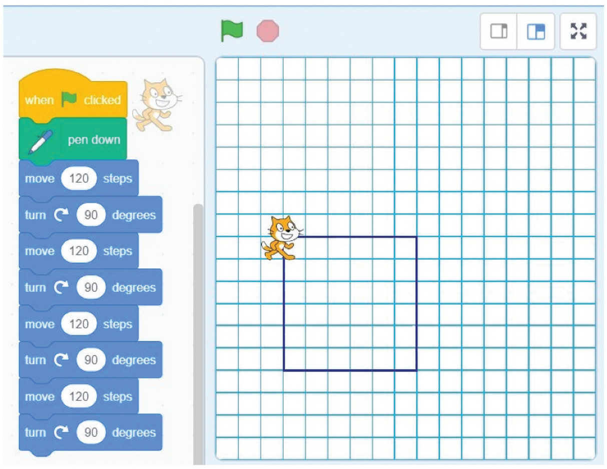
Task design

The task was designed in collaboration with the participating teacher. It entailed learning goals concerning geometry and programming and was created in line with Lithner’s (2017) criteria for CMR tasks – tasks for which the students need to construct (parts of) the method themselves. Since a CMR task requires students to create their own methods, the task needs to constitute a reasonable conceptual and creative challenge for the students. The students’ prior knowledge therefore needs to be considered. The participating teacher indicated that her students were all familiar with the concepts of squares and rectangles, a few knew how to calculate perimeters and areas, but none of them had been working with angles during mathematics lessons. With that as the starting point, three problems were designed (Figure 2). A successful solution to the first problem is shown in Figure 3. The problems aimed to give students an opportunity to use and develop their programming skills and engage in

Use Scratch to solve the following problems

1) Create a program that makes the cat draw a square.
2) Create a program that makes the cat draw another figure that is twice the size [area] of your square.
3) Create a program that makes the cat draw a triangle that has half the size [area] of the square that the cat drew in problem 1.

Figure 2. The problems used for the intervention; the concept of area was not used in the task.



```
when ⚑ clicked
pen down
move 120 steps
turn ↻ 90 degrees
move 120 steps
turn ↻ 90 degrees
move 120 steps
turn ↻ 90 degrees
move 120 steps
turn ↻ 90 degrees
```

Figure 3. Example of a program solving problem 1.

problem-solving. At the same time as the students solved the problems, they could gain an initial understanding of geometry concepts such as angle, perimeter, and area, however without introducing the concepts (that was done during the following lessons). Students were expected to realize, for example, that a "straight turn" or a "straight corner" [right angle] is 90 (degrees) and that doubling the "lengths of all sides" [perimeter] does not give a figure that is doubled in size [area]. The concept of degrees was new to almost all of the students and was introduced by Scratch as a "turning value" on the turning blocks (Figure 3). The design aimed to put reasonable demands on students' ability to handle Scratch. The students had an earlier lesson in which they learned how to choose, adjust, and merge moving and turning blocks to make the cat move, turn, and draw in Scratch. The students were expected to have the necessary programming skills to construct and manipulate geometric objects, to discern and adjust what is needed to solve the problem, and to make connections to formal mathematics. That is, they were expected to move from the *establishing practices mode* to enter the *critical discernment mode* and thereafter to the *situated discourse mode* engaging in reasoning that involves generalizing assumptions (Leung, 2011).

Teacher-student interaction

To prepare for teacher-student interaction supporting students' reasoning, the teacher and the first author of this paper anticipated possible reasoning paths, useful mathematics content, and programming skills the students were supposed to know in advance. Thereafter they prepared questions/suggestions to engage in teacher-student interaction following the earlier suggested principles of initiating, developing, verifying, and justifying their reasoning. The prepared questions to be used in the teacher-student interaction were formulated as general- and task-specific questions/suggestions, the latter to be asked if the general questions were not enough to support students' reasoning. Earlier studies have shown that task-specific questions are often formulated "in the moment" to suit the current situation, and if teachers are not prepared they often tend to provide the solution instead of challenging students' reasoning (Teledahl & Olsson, 2019). However, all teacher-student interactions are "in the moment" and the teacher sometimes needs to quickly adjust the prepared questions/suggestions or create new ones to fit the current situation yet still support CMR. The task-specific questions concern the problem's inherent mathematics and/or how to manage to program in Scratch. The aim of the prepared questions is to challenge students to use and take advantage of Scratch's potential and support the students to enter higher epistemic modes (Leung, 2011). Examples of questions and suggestions are presented in Table 1.

In the teacher's perspective, the interaction design is intended to inform the teacher about which principle the student needs support with. The teacher-student interaction furthermore needs to target the mathematics as well as the programming of the problem, especially the prior knowledge students need to solve the problem. Therefore, preparation for the teacher-student interaction included consideration of what knowledge was necessary to engage with, how to use that knowledge to develop their reasoning and justify their solutions, and how solutions (and parts of solutions) could be verified using Scratch. In summary, the interaction design can be described as follows:

- Ask general questions to elicit students' reasoning. If students are not expressing coherent reasoning (or no reasoning at all), ask the task-specific questions.
- If students' reasoning needs to be developed, ask general questions. If students are not expressing coherent reasoning, ask the task-specific questions.
- If students have reached a solution, ask general questions. If students are not expressing coherent reasoning, ask the task-specific questions.
- If students express incorrect reasoning or if they are not sure their solution is correct, ask general questions. If students do not suggest a way to verify their reasoning, ask the task-specific questions.

The ultimate goal for the interaction is for the student to reach a verified and justified solution.

Informants and data collection

Two grade 4 classes, altogether 46 students aged 10–11 years, were taught by the participating teacher, and 40 of the students volunteered to participate in the data collection. Written consent from the parents was collected before the intervention. The teacher had collaborated with the researchers for three years in a project about developing teaching that promotes CMR. The students were therefore familiar with solving problems and being asked to present and justify their solutions.

The intervention was conducted during two ordinary 60-minute math lessons, one class before lunch and the other class after lunch. The students were paired together sharing one computer. The students could call for the teacher whenever they needed to. A screen- and audio-recording software was installed on the participating students' computers. The participating teacher and the two researchers each carried their own recording device to ensure that all interactions with the students were recorded. Data consisted of 20 recordings (screen recordings and audio files). Students not participating in the data collection were paired together; they solved the same problems, but their screens and dialogs were not recorded.

Method of analysis

The analysis was conducted in the following steps:

(1) The recordings were divided into shorter sequences identified as situations including teacher-student interactions. The student-student interactions before and after the teacher-student interactions were also included.
(2) The situations were then categorized according to the principles: initiating, developing, verifying, or justifying reasoning. A fifth category concerned situations where none of the principles were identified.

(3) The next step was to examine whether or not the students' engaged in creative mathematical reasoning (CMR) during or after the teacher-student interaction. A reasoning sequence is identified as a result of creative mathematical reasoning when the statement is anchored in the inherent mathematics of the task (Lithner, 2008).
(4) As a final step, all interactions were analyzed to identify whether and how students used Scratch in any of the Leung's modes (2011) to initiate, develop, verify, and justify their creative mathematical reasoning.

Results

All student groups managed to solve problems 1 and 2 with or without support from a teacher-student interaction (including interactions with the researchers acting as teachers). Five groups of students started to work with problem 3, and three of them succeeded. Since the present study focuses on teacher-student interaction, the following will present the results of the analysis of situations where students asked for help or prepared for teacher-student interaction or when the teacher chose to initiate interaction. There were 55 situations altogether. These situations were categorized as mainly concerning any of the following principles: initiating (6 st), developing (20 st), verifying (6 st), or justifying (14 st) reasoning, and nine situations that were categorized as not belonging to any of the principles. The categorized situations were further analyzed to determine whether and how teacher-student interactions and the use of Scratch might have supported students to engage in CMR. The students had used Scratch during an earlier technology lesson and were expected to know how to choose, change, and add blocks to construct a program making the cat draw and move. That is, all students were expected to be prepared to move from the *establishing practices mode* (Leung, 2011) to the *critical discernment mode* and maybe even to enter the *situated discourse mode*. The results are presented below, one principle at a time, followed by the fifth category of situations not identified as addressing any of the four principles. All examples are chosen to be representative for each category of teacher-student interactions. Since the teacher mainly used the prepared questions/suggestions, the teacher-student interactions within each category were fairly similar.

Initiating reasoning

In six situations students were unable to initiate their reasoning. In three of these situations the teacher's general questions and suggestions (e.g., having the students reread the task) were sufficient to bring them to initiate CMR (See Appendix, section A for an example).

In three situations, general feedback did not help the students. The students in the example below needed a teacher-student interaction that included task-specific questions before they were able to express any reasoning at all and engage in CMR (Table 2).

Tim and Joe initiate their reasoning and create a program to make the cat draw a square (Table 3). Tim is one of the few students who had prior insights about the relation between the turning value 90 and a straight turn.

Summary: These students did not manage to initiate any reasoning on their own, and the teacher's general questions did not support them to. However, the teacher's task-specific questions prompted the students to activate prior knowledge about squares and how to use Scratch to make the cat move and draw. During the teacher-student interaction, Joe and Tim *initiated their reasoning* and eventually expressed *creative mathematical reasoning* anchored in intrinsic mathematics that squares have sides of equal length and 90 (degree) "turns." Their

Table 2. Teacher-student interaction to initiate reasoning.

Dialogue	Analysis
Tim: We don't understand. Teacher: Have you read the task? Joe: Yes . . . we are going to use Scratch to draw a square. But we do not know how to begin. Tim & Joe: (silence)	The teacher asks a prepared general question. The teacher interprets that the students know what the problem is asking them to do. The students do not initiate any reasoning and the teacher moves on to task-specific questions.
Teacher: What do you know about squares? Tim: Four sides that are the same. Teacher: The same? Joe: The sides have equal lengths, but . . . Tim & Joe: (silence)	The teacher asks a prepared task-specific question to initiate the students' reasoning by activating prior knowledge about squares. The teacher interprets that the students have an idea of what squares look like, but when they do not initiate any reasoning she chooses another task-specific question.
Teacher: What do you need to do to make a square? Tim: We need to make the cat walk and draw four equal sides . . .	The teacher asks two prepared task-specific questions to initiate her students' reasoning by activating prior knowledge about how to draw in Scratch.
Joe: . and to turn straight three, no four times. Teacher: Do you know how to do that?	The students initiate their reasoning, using their knowledge about squares and expressing what they need to do to solve the problem.
Joe: Yes, we have done that before.(The teacher leaves her students)	

choice to use their prior knowledge about Scratch moved them from *the establishing practices mode* to enter *the critical discernment mode* to create a program to verify their reasoning on how to make the cat draw lines and make 90-degree turns to draw a square.

Developing reasoning

We identified 20 situations as involving students who needed to develop their reasoning. In all of these situations the students needed task-specific feedback to develop their reasoning since they were not able to benefit from the teacher's general questions. Some of these were situations where students got stuck due to a mistake (17 st). In others, the teacher regarded the students' reasoning as insufficient and asked them to develop it (see Appendix, section B for an example). In two out of the 17 situations concerning mistakes the students did not engage in CMR. In these two interactions the teacher, after several attempts to use prepared questions/suggestions, instead created and gave task-specific suggestions that did not focus on their reasoning but rather how to solve parts of the task. For example. "You have to choose more steps to get the cat to move as you intended." However, in 18 of the 20 identified situations, students engaged in CMR after the teacher-student interaction. Below there is one example where the teacher-student interaction was needed to bring the students to develop their reasoning to correct a mistake.

Table 3. Students initiate their reasoning.

Tim: I think we take 60 steps on each side . . . then the cat will move over there (pointing).	The students engage in reasoning, choosing blocks to create a program drawing a square. Their reasoning is anchored in mathematics inherent to the problem.
Joe: [*Adds a block: Walk 60 steps*] . . . let's try [*Presses start*] . . . yes now she's moving . . .Now she needs to make a straight turn down.How much is a straight turn? (silence)	Tim activates his prior knowledge that a straight turn needs a turning value of 90 degrees.
Tim: I think that we need (silence) 90 to make a straight turn . . . and then 60 steps again.	The students verify their reasoning by adding blocks and running the program.
Joe: Okay, let's try. [*Adds two blocks: Turn 90, and Walk 60 steps. Presses start*]Yes! 90 will make her turn straight down.So, 60 steps and a 90-turn, four times . . .	They create a program that makes the cat draw the square and verify their reasoning by running the whole program twice.

Table 4. Students' reasoning not anchored in mathematics.

Dialogue	Analysis
Ann: Let's have 200 steps. Bea: That is not twice as much. Ann: Twice as much of what? Bea: The amount of steps . . . 180 + 180 . . . 360 steps. Ann: Ah, to make the cat move twice as far. Bea: Yes, to draw a side twice as long. [*Changing the value 180 to 360*]	The students' reasoning is anchored in mathematics. Twice as many steps will make the side twice as long.
Ann: And twice as much as 90 degrees is 180 degrees. Bea: Great. [*Changing the value 90 to 180, presses start*]	Their reasoning is not anchored in the mathematics concerning "turns" (angles). Their shallow reasoning pursues the idea that in order to make the figure twice the size everything needs to be doubled.

Developing reasoning to correct a mistake

These students have successfully solved the first problem. They constructed a square, each side 180 steps long, and while solving that problem they concluded that a square has 90-degree corners. In the following interaction they have taken on the second problem: construct a figure twice as large as the first one. They do not succeed, and when they do not discover their error they ask for help (Table 4).

The cat goes back and forth twice drawing a line, and then stops (Figure 4).

The students try to understand why the cat did not move as intended (Table 5).

Summary: These students, initiated reasoning that was incorrect and only partly anchored in mathematics. They came up with a method – double all values – but verified that as incorrect using Scratch. The teacher's task-specific questions, adjusted and created in the moment, encouraged them to *develop their reasoning* and compare what they intended the cat to do (make a straight turn) to what actually happened (turning around). The students activated prior insights they had gained when they

Table 5. Teacher-student interaction to develop reasoning.

Dialogue	Analysis
Bea: I don't get it, let's ask. Teacher: Hi, what's up? Ann: We meant to make a square twice as big but the cat just drew a line . . .	The students are not able to develop their reasoning. They ask for help.
Teacher: How did you reason when you constructed your program? Ann & Bea: (silence)	The teacher chooses to use and adjust one of the prepared general questions, asking them to narrate their reasoning. The students do not answer and she moves on to a task-specific question.
Teacher: What does your program tell the cat to do?	The teacher specifies her earlier questions to encourage the students to narrate their reasoning by asking them to relate their reasoning to the program they created.
Ann: We chose twice as many steps to make her walk twice as far . . . (pointing at the blocks and the corresponding moves) and twice as many turning degrees.	The students describe their reasoning, the first part anchored in mathematics, the second part based on the shallow reasoning that "everything needs to be twice as big."
Teacher: What happens then?	The teacher creates, in the moment, a task-specific question aiming to make them relate their programming to the cat's movements.
Bea: The cat turns twice as much. Teacher: Okay.	The teacher takes a step back, and the students develop their reasoning.
Bea: But that is not what we wanted. Ann: Ah . . . turning 180 degrees is the same thing as just turning around . . .	The students use the program to identify their mistake, recognizing what a 180-degree turn looks like.
Bea: Yes, she just runs back and forth. We need 90 here as well (as when they constructed a square) to make her turn straight. A straight (turn) is 90.	The students use the mathematics knowledge gained while constructing a square, that a straight turn is 90 degrees. The students develop their reasoning now anchored in mathematics.

Table 6. Teacher-student interaction to verify reasoning.

Dialogue	Analysis
Teacher: What have you done? Meg: We want the cat to turn straight down, but we do not know how much. Kim: We thought that turning 100 would work, but we do not know . . . (silence) Teacher: Have you tried your idea to see if it works? Meg: No, we did not . . . (silence) Teacher: Then I suggest you try.(The teacher leaves her students.)	The teacher asks a prepared general question to bring the students to narrate their reasoning, which might lead them to develop their reasoning. The students explain their problem, and one of them suggests an idea that they have not tried yet. The teacher asks a prepared general question to determine whether they have verified their idea or not.The teacher gives an adjusted prepared suggestion that the students need to verify their reasoning.
Kim: Okay, let's try; add 100 in the turning block. [*Adds 100 and presses start*] Meg: No, she turns too much . . . try 95. Kim: Okay. [*Changes to 95*] No, still too much, let's try 90. [*Changes to 90*] Meg: I can't see; make the cat walk another 30 steps. Kim: Sure. [*Adds a Walking block, 30 steps*] Meg: Yep, 90 it is! Kim: So, 90 will make her turn straight.	The students use Scratch to verify their idea. They systematically change the value of the turning block until the cat makes a straight turn. They add another block to make the verification easier to observe. They conclude that 90 will create a straight turn.

solved the first task (that the turning value of 90 made the cat turn straight) and changed 180 degrees to 90 degrees. During the teacher-student interaction, Ann and Bea developed their shallow reasoning – doubling all values – and eventually expressed *creative mathematical reasoning* anchored in intrinsic mathematics, stating that squares and rectangles have 90-degree "corners." These students moved from the *establishing practices mode* to enter the *critical discernment mode* using Scratch, creating a program to verify their ideas while initiating and developing their reasoning. They furthermore used Scratch as a base from which to generalize, making the conclusion that a "straight turn is always 90 degrees." That is, Bea and Ann entered the *situated discourse mode.*

Verifying reasoning

The data provide several situations where the students used Scratch to try out ideas to verify their reasoning without any teacher-student interaction. However, there are six situations where students have an idea, correct or incorrect, but they do not try to verify it until the teacher-student interaction requires them to. In two situations the students needed task specific questions (see Appendix, section C for an example). However, in four cases the students merely needed a general question or suggestion to resume their work and engage in CMR. The interaction with Meg and Kim is one example (Table 6).

Meg and Kim have created a program making the cat draw a straight line 30 steps long. They have an idea about how to continue to make the cat turn, but they are unsure and ask for help.

Summary: These students engaged in reasoning and came up with an idea about how to draw a square, but they were not sure that their reasoning was correct. However, they did not attempt to verify whether their idea was correct. The teacher asked a general question to bring them to *verify* their idea. During the teacher-student interaction, Meg and Kim verified their reasoning as incorrect and used Scratch to develop their *creative mathematical reasoning* anchored in intrinsic mathematics that 90 degrees will make a straight turn. These students moved from the *establishing practices mode* to enter the *critical discernment mode* using Scratch to verify their ideas.

Table 7. Teacher-student interaction to justify reasoning.

Dialogue	Analysis
Mia & Kate: Look, we made it. Teacher: Ok, are you sure you have a square?	The teacher does not confirm the students' solution. Instead, she asks a prepared task-specific question, encouraging them to justify why the drawn figure is a square.
Kate: First we created that block (pointing at the first move and turning block) 180 steps and 90 turn and then we had to go another 180 steps for the sides to be equally long. Mia: And then we turned 90 and moved 180 steps. Teacher: Why did you have 180 steps on each side?	The students use the program blocks to explain their reasoning and the teacher creates and asks a task-specific question.
Kate: Because in a square all sides are equally long.	The students' answer contains a part of the definition for a square, which is anchored in mathematics.

Justifying reasoning

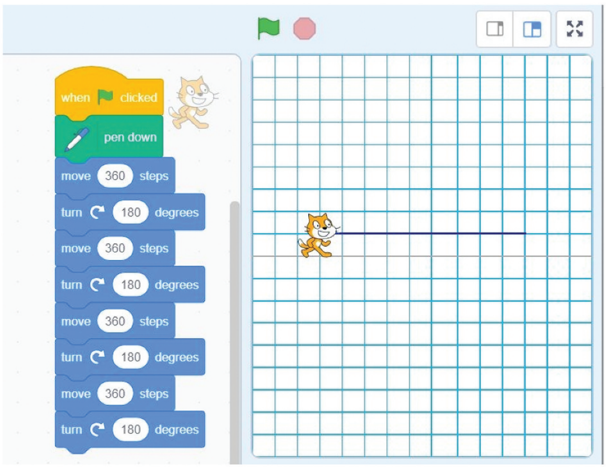
The data show that students rarely try to justify their reasoning even though they have solved the given problem and are convinced that their reasoning is correct since the cat draws what they expected. The teacher is well aware of this and takes the opportunity to engage in a teacher-student interaction encouraging the students to justify their reasoning. In seven situations the teacher's general question (e.g., asking them to explain why their solution is correct) brought the students to justify their work, while in four situations the teacher needed to ask task-specific questions. In all these 11 situations the students managed to anchor their justifications in intrinsic mathematics, a key to engage in CMR. One example of the latter is an interaction with Mia and Kate. They have solved the first problem, successfully creating a program to draw a square (Table 7).

Summary: These students had initiated and verified their reasoning as correct using Scratch. However, they had not *justified their reasoning* and the teacher chose a task-specific question to have them do so. During the teacher-student interaction, Kate and Mia verified their *creative mathematical reasoning* by formulating arguments anchored in mathematics explaining why their figure was a square. These students moved from the *establishing practices mode* to enter the *critical discernment mode* using Scratch to verify their ideas.

However, since all of her students are accustomed to the teacher asking for justifications, there are three situations where students aim to justify their reasoning to prepare for a teacher-student interaction. After a fair amount of exploration and unsuccessful attempts, Ella and Nora have successfully drawn a square and a rectangle twice as large as the square. While constructing the square, they explored to determine the proper "turning value" and finally concluded that a straight turn is 90 degrees. When creating the rectangle, they made

Table 8. Student-student interaction preparing to justify reasoning.

Dialogue	Analysis
Ella: We need to tell her (the teacher). Nora: That we are done?	Ella and Nora start to prepare for the interaction with their teacher so they can answer the questions they know she will ask.
Ella: No, why this one (the rectangle) is twice as big as our square. Nora: Well, but that is why we chose twice as many steps. The cat needs to walk twice as much to make the sides double.	Based on their earlier exploration and attempts to determine the length of the sides, they create arguments for why making two sides twice as long will create a rectangle twice as big.
Ella: That will do . . . but only two sides . . . otherwise it became too large. Nora: Of course, we need a rectangle that is as big as two squares or how to put it . . . Nora: We needed 90 turning blocks here as well. Ella: Yes, otherwise (using 180) the cat just turns around . . . a straight turn up or down is always a 90 turn, so 90. Nora: Okay, now we know, let's ask her (the teacher) to come.	Based on their earlier exploration and attempts to decide how to choose values to make a straight turn, they have concluded that a straight turn is always 90 degrees.



```
when clicked
pen down
move 360 steps
turn 180 degrees
move 360 steps
turn 180 degrees
move 360 steps
turn 180 degrees
move 360 steps
turn 180 degrees
```

Figure 4. The students' programming making the cat draw a line.

the same assumptions as Ann and Bea (Table 4 & 5); they chose all values twice as large. But after some troubleshooting they decided that they needed to make just two sides twice as long and to continue to choose 90 degrees. The following is a situation where they prepare for an interaction with their teacher justifying their reasoning (Table 8).

Summary: These students initiated, developed, and verified their reasoning and managed to construct a square as well as a rectangle twice as large as the square. They had not yet *justified* their reasoning, however. To prepare for the teacher-student interaction, Ella and Nora worked to justify their reasoning so they would be able to answer the expected questions from the teacher. They used their earlier troubleshooting and referred to their programming when they created arguments anchored in mathematics to justify their *creative mathematical reasoning*. These students moved from the *establishing practices mode* to enter the *critical discernment mode* using Scratch to verify their ideas. During their troubleshooting they also entered the *situated discourse mode* when they used Scratch to generalize the assumption that "straight turns" are always 90 degrees.

Providing feedback without supporting students' reasoning

In nine situations, the teacher-student interaction did not address any of the four principles. In five situations the difficulty concerned the software, such as how to Zoom, move the "workspace," how to choose "editing mode" etc. Two situations concerned how to choose programming blocks to make the cat draw. When the students addressed their teacher with such difficulties, the teacher gave direct instructions for overcoming the problem. There were also two situations where the teacher-student interaction could have addressed the principle of *developing* their reasoning. However, in both situations the researcher acting as teacher chose to explain what to do rather than supporting the students' reasoning. That is, the students did not engage in CMR and found themselves having the same difficulty solving the next part of the task (see Appendix, section D for an example).

Conclusion

RQ 1: How can teacher-student interaction support students to initiate and develop their creative mathematical reasoning?

The study showed that the teacher-student interactions initiated by the prepared questions targeting students' reasoning provided the teacher with information about the difficulties students encountered concerning any of the four principles. With general questions, the teacher-student interaction addressed the students' interpretation and understanding of the task. Those questions prompted the students to reread the text, giving them an opportunity to interpret it once more, restate their interpretation, or express their initial reasoning verbally. Being asked to examine the problem once more led some of the students to *initiate* their reasoning. If not, the task-specific questions helped the students sort through, select, and focus on useful task-specific concepts. They prompted the students to activate useful prior knowledge and focus their reasoning on the intrinsic mathematics of the task (e.g., the characteristics of a square) and on the programming blocks needed to solve problems such as how to make the cat move and so forth. Merely activating useful prior knowledge brought the students to *initiate* or, when necessary, to *develop* their reasoning. The study furthermore showed that to *develop* their reasoning students were dependent on task-specific questions/suggestions that encouraged them not only to focus on useful prior knowledge but also to discover and correct mistakes, generate ideas, and/or anchor their reasoning in intrinsic mathematics.

RQ 2: How can teacher-student interaction support students to justify and verify their creative mathematical reasoning?

A vast majority of the students used Scratch to verify their reasoning by themselves. However, when the teacher realized that her students did not trust their reasoning to be correct, general questions such as "Have you tested your idea to see if it works?" initiated teacher-student interaction that prompted the students to use Scratch to verify whether their reasoning was correct. The students essentially tried to justify their reasoning when the teacher specifically asked them to or as preparation for an interaction with the teacher (knowing that the teacher would ask for a justification). Being asked or preparing for questions such as "Are you sure that you have drawn a square?" challenged the students to look for arguments anchored in the mathematics and their programming of the task to justify their reasoning.

RQ 3: How do students make use of Scratch to engage in creative mathematical reasoning?

A full engagement in CMR means constructing a new (for the student) reasoning sequence and formulating arguments anchored in mathematics explaining why the solution method works and why the solution is correct. Such reasoning sequences will entail all four principles: initiating, developing, verifying, and justifying reasoning. To *initiate* their reasoning, when creating a program to draw a square, for example, the students used their prior programming knowledge to create an idea of how to choose the turning-block value to make the cat make a straight turn (e.g., "Let's try 100 as turning value"). Scratch would thereafter *verify* their initial reasoning as correct or (as in most cases in this study) incorrect.

To *develop* their reasoning, the students continuously used Scratch's instantaneous visualization of their ideas to make necessary adjustments (e.g., "A turning value of 100 makes the cat turn too much. Let's try 60") and thereafter to verify their adjusted idea. As they developed their reasoning, students formulated arguments on how and why their ideas worked (or did not work) and eventually approached the correct programming. When the students *justified* their reasoning, usually when asked by the teacher to do so, they often referred to the documentation of their activities in Scratch.

Discussion

Problem-solving through CMR is described as beneficial for learning mathematics; however, it has also been pointed out as challenging (e.g., Jonsson et al.,(2016).; Lithner, 2017; Olsson & Granberg, 2019). Because programming is a problem-solving activity in itself, using it to solve a mathematics problem adds to that challenge and might even hamper students' learning of mathematics (Brennan & Resnick, 2012; Resnick et al., 2009). These findings indicate a need for teacher support. Lithner (2017) proposes an overarching strategy to support CMR: to identify the difficulties students encounter during problem-solving and provide formative feedback supporting students' ability to construct a solution. These proposals are very general and provide no clear guidance for teachers when it comes to *how* to identify task-specific difficulties and *how* to formulate formative feedback that will target the students' reasoning. Earlier studies (D'Arcy & Olsson, 2021); Olsson & Teledahl, 2018) have suggested four principles (see *Framework*) focusing on students' reasoning and guiding teachers regarding how to support students' reasoning trajectories. In the present study we have further developed the didactic design based on these principles to include general questions to elicit students' reasoning and identify their difficulties as well as task-specific questions and suggestions to support students' reasoning according to each of the principles (see *Method* Table 1). These questions and suggestions aim to target students' reasoning with respect to the mathematics as well as the programming included in the problem. The results, how teacher-student interaction focusing on students' reasoning could reduce the complexity of problem-solving and programming, are discussed below. The discussion ends with some implications for teaching.

Supporting students' programming and creative mathematical reasoning

The present study showed that teacher-student interaction initiated by asking general questions supported the teacher in identifying students' difficulties as related to any of the principles of the hypothetical reasoning trajectory (Table 1). The principles are initiating reasoning, developing reasoning, verifying reasoning as correct or incorrect, and justifying reasoning (see *Results*). Students' shortcomings in *initiating* and *developing* reasoning could to some extent be attributed to the complexity of simultaneously solving mathematical tasks and engaging in programming and thus having to identify and use appropriate prior knowledge of both (Barr & Stephenson, 2011). The task-specific questions used in the present study were shown to help students recognize and activate useful knowledge in mathematics as well as programming (see *Results* Tim & Joe). These questions could be described as one way of providing metacognitive feedback since the interaction supported the students in making decisions to select and implement resources or strategies learned about earlier. The latter is described as "metacognition" and as a crucial problem-solving competence (Schoenfeld, 1985). However, providing metacognitive feedback presupposes that students have prior mathematical knowledge and programming skills to activate. The students need sufficient prior knowledge to understand the problem and the necessary skills to interact with the technology, as Leung (2011) describes in terms of being in the *establishing practices mode.*

To *develop* their reasoning, students may need to gain new insights into mathematics. This occurred, for example, when students were dealing with an incorrect assumption (see *Results* Ann & Bea) or exploring the mathematical relationships of the task to determine how many degrees the cat needs to turn to make a straight turn. The students used their programming to *verify* (parts of) their reasoning as correct or incorrect while gaining insights into the mathematical relation between 90 degrees and "straight turns" (see *Results* Tim & Joe). Even though the students did not need any new programming skills to do so, they did need to find out how to *make use* of the programming. That is, they had to go from just getting the cat to move to making active choices about how to use mathematics and programming blocks to find out how to get the cat to move as intended. This transition is described by Leung (2011) as entering the *critical discernment mode*. The students needed to identify variations and patterns of the cat's movements, gather information, and draw conclusions from that. Leung (2011) suggests that the transitions between the modes entail cognitive effort. Some of the students in the present study needed the teacher's encouragement to actually try to use Scratch to verify their reasoning or they needed the teacher's suggestions on how to use their programming to get the information they needed (see *Results* Anne & Lisa). These situations exemplify how teachers can use Scratch to support students to be active learners (Mayer, 2014). In dialogue with the teacher, students can come up with actions in Scratch and receive immediate feedback from the software, whereafter the teacher may ask the students how to interpret the feedback (Moreno & Mayer, 2007).

However, even when students have *initiated*, *developed*, and *verified* their creative reasoning as correct and thereby solved the problem, they might not have anchored their reasoning in the intrinsic mathematics of the problem. For example, they might have used the technology to engage in a trial-and-error strategy to solve the problem instead of engaging in any deeper reflection. In such cases their reasoning cannot yet be described as CMR according to Lithner (2008). To engage in CMR they need to *justify* their reasoning, presenting mathematically anchored arguments for why their method works and sometimes developing their reasoning to match their actual programming. Earlier research has found that students are not likely to engage in justifying their solutions by themselves (Ball & Bass, 2003). The general and task-specific questions targeting students' justifications were shown to challenge the students to find arguments anchored in the mathematics as well as their programming (see *Results* Mia & Kate). As described earlier, the students in this study were used to the teacher asking them to justify their solutions, so there were also examples of students preparing their justification on their own (see *Results* Ella & Nora). Regardless of whether students managed to come up with a justification during the teacher-student interaction or while preparing for one, they used their programming to anchor their arguments in mathematics. They developed their reasoning to include generalizations concerning, for example, the relation between the value they entered in their programming and the shape of the turn the cat made. Following Leung's (2011) framework, this could be described as moving from the *critical discernment mode* to enter the *situated discourse mode*, a transition that is also likely to require a cognitive effort from the students. Hence, it is not surprising that most of the students needed the teacher's expectations of justification and/or support to provide it.

All students in the present study gained and generalized the insight that a straight turn – angle – always needs to be 90 degrees; however, several of the students did not come to this conclusion while programming without teacher support. The present study confirmed Benton et al.'s (2018) arguments that students do not automatically learn mathematics while programming. For example, it may be that the programming itself hampered the mathematical reasoning of students who were unable to transfer from the establishing practices mode to either of the other modes (see Results Tim & Joe). Even if students entered the *critical discernment mode*, Scratch allows them to use trial-and-error strategies to try solving the problem without deeper reflection. This is also an example of how technology can hamper mathematical learning.

The present study has shown that students can learn mathematics using Scratch when they are supported by a teacher-student interaction targeting their reasoning so that they enter the *critical discernment mode* and thereafter the *situated discourse mode* (Leung, 2011). During the interaction the students engage in CMR (Lithner, 2008) by reflecting, generalizing, and creating mathematically anchored arguments supporting their programming. These results are in line with Ball and Bass (2003) and Olsson (2019), who suggested that justifications are keys to deeper knowledge, although they acknowledged that it is a challenge for teachers to bring them about. Moreover, earlier research suggests that students deepen their mathematical knowledge and understanding when solving tasks using Scratch (Calao et al., 2015; N. Calder, 2018; Kong & Kwok, (2022).2) and that they also improve their reasoning skills (Calao et al., 2015). We suggest that teacher-student interactions encouraging mathematical justifications can be a vehicle for students to transit into *critical discernment* and *situated discourse modes* and that the interactions thereby support students to deepen their knowledge and understanding by CMR.

It was furthermore noted that when students transit from the *establishing practices mode* and enter either of Leung's higher modes, the *critical discernment* or *situated discourse mode*, the programming could support rather than hamper their reasoning. This occurred, for example, in situations where students used Scratch to visualize and verify their reasoning. The students also referred to their programming activities when they explained or justified their reasoning to each other and to the teacher.

Implications for teaching mathematics and programming

As described earlier, we have been inspired by Simon's (1995) model of hypothetical learning trajectories. Simon (Ibid.) suggests that the teacher needs to anticipate and acknowledge students' hypothetical learning trajectory while teaching and supporting students to reach the learning goals. In our adaptation of Simon's model we consider students' reasoning as a prerequisite for their learning, and consequently the teacher needs to anticipate students' hypothetical reasoning trajectory: initiating, developing, verifying, and justifying reasoning.

Using Scratch to teach mathematics creates a complex learning situation. For the students to engage in problem-solving, and specifically in CMR, the teacher needs to consider the didactic design, choosing a learning goal that suits the students' prior mathematical knowledge and finding a problem targeting that learning goal. Furthermore, the problem must be one that can be solved using Scratch, and the students must have enough programming knowledge to work with the problem. In other words, the students need to be in Leung's *establishing practices mode*, knowing how to make the cat move, turn, and draw. During preparation, the teacher will have identified what prior knowledge students need to activate and use when solving the given problem. Thereafter the teacher needs to anticipate students' hypothetical reasoning trajectory and prepare general questions to elicit their reasoning and target any shortcomings with respect to any of the reasoning principles. The teacher also needs to prepare task-specific questions and suggestions to give timely support to students to activate appropriate prior knowledge, verify their reasoning, and so forth depending on the phase of the reasoning trajectory with which the students need help.

Of course it takes both thorough preparation and extensive practice to develop skills to determine in a timely manner how to support students' creative reasoning. That is illustrated in the example (see Appendix, section D) where the students ask one of the researchers for help. Even though the researcher had made preparations in collaboration with the teacher, he failed to support the students as intended since he guided them rather than asking any of the well-prepared questions. However, the teacher almost always managed to provide timely support to students by initiating teacher-student interactions as she had planned. Turning back to the challenges involved in bringing students to justify their reasoning (e.g., Ball & Bass, 2003) and to learn mathematics while programming (e.g., Benton et al., 2018), it is worth noting that the teacher in the present study, collaborating with the first author, has developed her skills to prepare and carry out interactions to support her students' reasoning trajectories over a fairly long time. However, this article reports on the first time she introduced Scratch in a mathematics lesson, which indicates that the didactic design, based on the four principles targeting students' reasoning, is transferable to other teaching situations.

Acknowledgments

We thank ***Denice D'Arcy*** for her advices as expert teacher when designing the intervention.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Carina Granberg http://orcid.org/0000-0002-5607-213X

Additional information

Notes on contributors

Jan Olsson is a senior lecturer at university of Mälardalen, Sweden. He is also a member of Umeå Mathematics Educational Research Center (UMERC), Umeå university. His research interest is teacher-student interactions aiming at students' learning of mathematics by reasoning. The last six years he has been involved in research in close collaboration with professional teachers.
Carina Granberg, PhD, is Assistant professor at the Department of Applied Educational Science at Umeå University, Sweden, and is a member of Umeå Mathematics Education Research Centre (UMERC). Her main research interests are Problem solving, Mathematical reasoning, Programming and Dynamic software

References

Ainsworth, S.E., Bibby, P.A., and Wood, D.J. (1998). Analyzing the costs and benefits of multi-representational learning environment. In Someren, M. W. van, Reimann, P. (Eds). *Learning with Multiple Representations* (Oxford, U.K.: Elsevier Science) (pp. 12–134)

Balanskat, A., & Engelhardt, K. (2014). *Computing our future: Computer programming and coding-Priorities, school curricula and initiatives across Europe*. European Schoolnet.

Ball, D., & Bass, H. (2003). Making mathematics reasonable in school. In J. Kilpatrick, G. Martin, & D. Schifter (Eds.), *A research companion to principles and standards for school mathematics* (pp. 27–44). National Council of Teachers of Mathematics.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? *Acm Inroads*, *2*(1), 48–54. https://doi.org/10.1145/1929887.1929905

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of child-computer Interaction*, *16*, 68–76. https://doi.org/10.1016/j.ijcci.2017.12.004

Bhagat, K.K., and Chang, C.Y. (2015). Incorporating GeoGebra into Geometry learning-A lesson from India. *Eurasia Journal of Mathematics, Science and Technology Education* 11 (1) (pp. 77–86) doi:https://doi.org/10.12973/eurasia.2015.1307a

Boaler, J. (2002). The development of disciplinary relationships: Knowledge, practice and identity in mathematics classrooms. *For the Learning of Mathematics*, *22* (1), 42–47. https://www.jstor.org/stable/40248383

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

Brown, Q., Mongan, W., Kusic, D., Garbarine, E., Fromm, E., & Fontecchio, A. (2008). Computer aided instruction as a vehicle for problem-solving: Scratch boards in the middle years classroom. In *2008 Annual Conference & Exposition* (pp. 13–319).

Buckley, B. C., Gobert, J. D., Kindfield, A. C., Horwitz, P., Tinker, R. F., Gerlits, B., Willett, J., Dede, C., & Willett, J. (2004). Model-based teaching and learning with BioLogica™: What do they learn? How do they learn? How do we know? *Journal of Science Education and Technology*, *13*(1), 23–41. https://doi.org/10.1023/B:JOST.0000019636.06814.e3

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing mathematical thinking with Scratch. In G. Conole, T. Klobučar, C. Rensing, J. Konert, & E. Lavoué (Eds.), *Design for teaching and learning in a networked world. EC-TEL 2015. Lecture notes in computer science*, (Vol. 9307, pp. 17-27). Springer 17–27 . https://doi.org/10.1007/978-3-319-24258-3_2

Calder, N. S. (2009). Visual tensions when mathematical tasks are encountered in a digital learning environment. In M. Tzekaki, M. Kaldrimidou, & C. Sakonidis (Eds.), In *search of theories in Mathematics Education*, Proceedings of the 33rd annual conference of the International Group for the Psychology of Mathematics Education. Athens: PM

Calder, N. (2018). Using Scratch to facilitate mathematical thinking. *Waikato Journal*, *23*(2), 43–58. https://doi.org/10.15663/wje.v23i2.654

Clark-Wilson, A., Aldon, G., Cusi, A., Goos, M., Haspekian, M., Robutti, O.,& Thomas, M. O. J. (2014). The challenges of teaching mathematics with digital technologies—the evolving role of the teacher. In P. Liljedahl, C. Nichol, S. Oesterle, & D. Allan (Eds.), *Proceedings of the joint meeting of PME 38 and PME-NA 36* (Vol. 1, pp. 87–116). Vancouver: University of British Columbia. https://discovery.ucl.ac.uk/id/eprint/1473704/

Clark-Wilson, A., Robutti, O., & Thomas, M. (2020). Teaching with digital technology. *ZDM—The International Journal on Mathematics Education*, *52*(7), 1223–1242. https://doi.org/10.1007/s11858-020-01196-0

D'Arcy, D., & Olsson, J. (2021) *Kreativa resonemang i matematikundervisningen: Forskningsrön i praktiken* (Lund: Studentlitteratur AB). [Creative reasoning in mathematics teaching: Research in practice].

Drijvers, P. (2020). Digital tools in Dutch mathematics education: A dialectic relationship. In M. Van den Heuvel-Panhuizen, M (Ed.), *National reflections on the Netherlands didactics of mathematics* (pp.177–195). Springer. https://doi.org/10.1007/978-3-030-33824-4_10.

Erol, O., & Çırak, N. S. (2021). The effect of a programming tool scratch on the problem-solving skills of middle school students. *Education and Information Technologiess* 27 (October), 4065–4086 . https://doi.org/10.1007/s10639-021-10776-w

Forsström, S. E., & Kaufmann, O. (2018). A literature review exploring the use of programming in mathematics education. *International Journal of Learning, Teaching and Educational Research*, *17* (12), 18–32. http://hdl.handle.net/11250/2599710

Franke, M. L., Webb, N. M., Chan, A. G., Ing, M., Freund, D., & Battey, D. (2009). Teacher questioning to elicit students' mathematical thinking in elementary school classrooms. *Journal of Teacher Education*, *60*(4), 380–392. https://doi.org/10.1177/0022487109339906

Granberg, C., & Olsson, J. (2015). ICT-supported problem-solving and collaborative creative reasoning: Exploring linear functions using dynamic mathematics. *Journal of Mathematical Behaviour*, *37*, 48–62.https://doi.org/10.1016/j.jmathb.2014.11.001.

Granberg, C. (2016). Discovering and addressing errors during mathematics problem-solving—A productive struggle? *The Journal of Mathematical Behavior*, *42*, 33–48. https://doi.org/10.1016/j.jmathb.2016.02.002

Hiebert, J., & Grouws, D. A. (2007). The effects of classroom mathematics teaching on students' learning. In I. F. K. Lester (Ed.), *Second handbook of research on mathematics teaching and learning*. (Volume 1, pp. 371-404) Information Age Publishing 371–404 .

Hillmayr, D., Ziernwald, L., Reinhold, F., Hofer, S. I., & Reiss, K. M. (2020). The potential of digital tools to enhance mathematics and science learning in secondary schools: A context-specific meta-analysis. *Computers & Education*, *153*, 103897. https://doi.org/10.1016/j.compedu.2020.103897

Hoyles, C., & Noss, R. (2003). What can digital technologies take from and bring to research in mathematics education? In A. J. Bishop, M. A. Clements, C. Keitel, J. Kirkpatrick, & F. Leung (Eds.), *Second international handbook of mathematics education* (Vol. 1, pp. 323–349). Kluwer Academic.

Jonsson, B, Kulaksiz, B., and Lithner, J. (2016). Creative and algorithmic mathematical reasoning: effects of transfer-appropriate processing and effortful struggle. *International Journal of Mathematical Education in Science and Technology* 47 (8) (pp. 1206–1225) doi:https://doi.org/10.1080/0020739X.2016.1192232

Jonsson, B., Norqvist, M., Liljekvist, Y., & Lithner, J. (2014). Learning mathematics through algorithmic and creative reasoning. *The Journal of Mathematical Behavior*, *36*, 20–32. https://doi.org/10.1016/j.jmathb.2014.08.003

Kalelioglu, F., and Gülbahar, Y. (2014). The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective. *Informatics in education* 13 (1) 33–50 http://www.mii.lt/informatics_in_education/

Kong, S.C., and Kwok, W.Y. (2022). From mathematical thinking to computational thinking: Use scratch programming to teach concepts of prime and composite numbers. *In Proceedings of 29th International Conference on Computers in Education Conference. Asia-Pacific Society for Computers in Education.* Thailand, November (pp. 549–558) doi: https://doi.org/10.1002/int.22931

Kynigos, C., & Grizioti, M. (2018). Programming approaches to computational thinking: Integrating Turtle geometry, dynamic manipulation and 3D Space. *Informatics in Education*, *17*(2), 321–340. https://doi.org/10.15388/infedu.2018.17

Leung, A. (2011). An epistemic model of task design in dynamic geometry environment. *ZDM—The International Journal on Mathematics Education*, *43*(3), 325–336. https://doi.org/10.1007/s11858-011-0329-2

Lithner, J. (2008). A research framework for creative and imitative reasoning. *Educational Studies in Mathematics*, *67*(3), 255–276. https://doi.org/10.1007/s10649-007-9104-2

Lithner, J. (2017). Principles for designing mathematical tasks that enhance imitative and creative reasoning. *ZDM—The International Journal on Mathematics Education*, *49*(6), 937–949. https://doi.org/10.1007/s11858-017-0867-3

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, *41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Lye, S. Y., & Koh, J. H. L. (2018). Case studies of elementary children's engagement in computational thinking through scratch programming In M. Khine, (Ed.), *Computational thinking in the STEM disciplines* (pp. 227–251). Springer. http://doi.org/10.1007/978-3-319-93566-9_12

Martin, A., & Grudziecki, J. (2006). DigEuLit: Concepts and tools for digital literacy development. *Innovation in Teaching and Learning in Information and Computer Sciences*, *5*(4), 249e267. https://doi.org/10.11120/ital.2006.05040249

Mayer, R.E. (2014). Incorporating motivation into multimedia learning. *Learning and instruction* 29 (pp. 171–173) doi: https://doi.org/10.1016/j.learninstruc.2013.04.003

Misfeldt, M., & Ejsing-Duun, S. (2015). Learning mathematics through programming: An instrumental approach to potentials and pitfallsIn K. Krainer & N. Vondrová (Eds.), *CERME 9: Ninth Congress of the European Society for Research in Mathematics Education* 2524–2530. Prague, Czech Republic: Charles University in Prague, Faculty of Education and ERME. https://hal.archives-ouvertes.fr/hal-01289367

Moreno, R., and Mayer, R. (2007). Interactive multimodal learning environments. *Educational psychology review* 19 (3) (pp. 309–326)

Moreno-León, J., & Robles, G. (2016). *Code to learn with Scratch? A systematic literature review*. Prague, Czech Republic: Charles University in Prague, Faculty of Education and ERME April 2016 150–156 Abu Dhabi.

Norqvist, M., Jonsson, B., Lithner, J., Qwillbard, T., & Holm, L. (2019). Investigating algorithmic and creative reasoning strategies by eye tracking. *The Journal of Mathematical Behavior*, *55*, 100701. https://doi.org/10.1016/j.jmathb.2019.03.008

Olsson, J., & Teledahl, A. (2018). Feedback to encourage creative reasoning. In J. Häggström, Y. Liljekvist, J. Bergman Ärlebäck, M. Fahlgren, & O. Olande (Eds.), *Perspectives on professional development of mathematics teachers. Proceedings of MADIF 11, (pp. 141-150).* Gothenburg, Sweden: SMDF.

Olsson, J., & Granberg, C. (2019). Dynamic software, task solving with or without guidelines, and learning outcomes. *Technology, Knowledge and Learning*, *24*(3), 419–436. https://doi.org/10.1007/s10758-018-9352-5

Olsson, J. (2019). Relations between task design and students' utilization of GeoGebra. *Digital Experiences in Mathematics Education*, *5*(3), 223–251. https://doi.org/10.1007/s40751-019-00051-6

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Kafai, Y., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60–67. https://doi.org/10.1145/1592761.1592779

Sacristán, A. I., Calder, N., Rojano, T., Santos-Trigo, M., Friedlander, A., Meer, H., & Perrusquía, E. (2009). The influence and shaping of digital technologies on the learning–and learning trajectories–of mathematical concepts. In C. Hoyles, & J. B. Lagrange, (Eds.), *Mathematics education and technology-rethinking the terrain* (pp. 179-226). Springer.

Schoenfeld, A. (1985). *Mathematical problem-solving*. Academic.

Schoenfeld, A. H., & Sloane, A. H. (Eds.). (2016). *Mathematical thinking and problem-solving*. Routledge.

Simon, M. A. (1995). Reconstructing mathematics pedagogy from a constructivist perspective. *Journal for Research in Mathematics Education*, *26*(2), 114–145. https://doi.org/10.2307/749205

Sjöberg, C., Nouri, J., Sjöberg, R., Norén, E., & Zhang, L. (2018, July). Teaching and learning mathematics in primary school through Scratch. In *International Conference on Education and New Learning Technologies, EDULEARN18 Proceedings* (pp. 5625–5632).

Teledahl, A., and Olsson, J. (2019). Feedback for creative reasoning. In U. T. Jankvist, M. van den Heuvel-Panhuizen, & M. Veldhuis (Eds.), *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education*(pp. 3794-3801). Utrecht, The Netherlands: Freudenthal Group & Freudenthal Institute, Utrecht University and ERME. https://hal.archives-ouvertes.fr/hal-02430229

Vatansever, Ö., & Baltacı Göktalay, S. (2018). How does teaching programming through Scratch affect problem-solving skills of 5th and 6th grade middle school students? *Journal of Eurasia Social Sciences*, *9*(33), 1778–1801.https://doi.org/10.18844/wjet.v12i4.5179 .

Voogt, J., & Pelgrum, H. (2005). ICT and curriculum change. *Human Technology*, *1*(2), 157–175. https://doi.org/10.17011/ht/urn.2005356

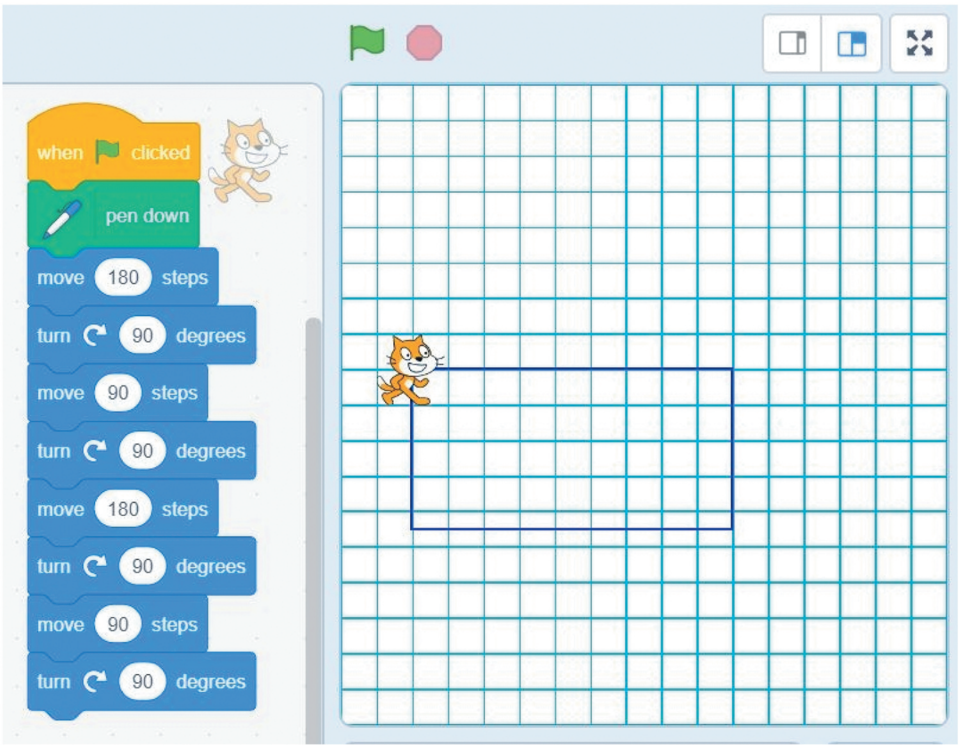
Weintrop, D., and Weintrop, U. (2015). To block or not to block, that is the question: students' perceptions of blocks-based programming. *In Proceedings of the 14th international conference on interaction design and children.* June Aarhus, Denmark (pp. 199–208) doi:https://doi.org/10.1145/2771839.2771860

Zhong, B., & Xia, L. (2018). A systematic review on exploring the potential of educational robotics in mathematics education. *International Journal of Science and Mathematics Education 18*)1) , 1–23. https://doi.org/10.1007/s10763-018-09939-y

(2014).Appendix

(A) ***Teacher-student interaction – Initiating reasoning***

Two students who did not initiate any reasoning at all. The teacher used general questions and suggestions to support the students' reasoning.
Table A. Teacher-student interaction to initiate reasoning



```
when [flag] clicked
pen down
move 180 steps
turn ↻ 90 degrees
move 90 steps
turn ↻ 90 degrees
move 180 steps
turn ↻ 90 degrees
move 90 steps
turn ↻ 90 degrees
```

Figure A1. The second task, create a rectangle twice as large as their earlier-constructed square.

Dialogue	Analysis
Jim: I don't get it! Teacher: What are you supposed to do? Jim: Make the cat draw something. Teacher: Just anything . . . hmm . . . Read the instructions out loud.Neo: (Reading out load) . . . a square? Jim: Ah . . . Like four equal sides.Neo: Sorry – I knew, I just forgot. The teacher leaves the students.	The teacher asks a prepared general question and interprets their answer as not using all given information. The teacher gives a prepared general suggestion – to read out load. Doing so, the students notice the missed information and activate prior knowledge concerning squares.
Jim and Neo initiate their reasoning, aiming to create a program to make the cat draw a square. They came up with and tested several ideas on how to choose a "turning value" before discovering that 90 (degrees) will make the cat make a straight turn.	

Summary: These students did not manage to initiate any reasoning on their own. The teacher's general questions prompted them to reread the task, notice information they had missed, and activate prior knowledge about squares. Once they understood the task and activated the necessary prior knowledge, these students used Scratch to explore, verify, and eventually gain insights about the relation between turning values and straight turns. In other words, they managed to express a creative mathematical reasoning and to move from *the establishing practices mode* to enter *the critical discernment mode*.

(B) ***Teacher-student interaction – Developing reasoning at the teacher's request***

The students in the following example successfully solved the first problem, gaining the insight that a straight turn is 90 degrees. They also solved the second problem (Figure 1). However, the teacher thought their reasoning was insufficient and prompted them to develop their reasoning.

Table B. Teacher-student interaction to develop reasoning

Dialogue	Analysis
Teacher: Hi, so have you solved that problem? Cia: We think this is twice as large as our square. Teacher: Tell me, what is twice as big?	The teacher finds that their reasoning: – "We think that it is twice as large" – is not anchored in mathematics and asks an adjusted prepared task-specific question prompting them develop their reasoning.
Cia: Our square was this big (drawing a square half the size of the rectangle).	The students compare the area of the square to the rectangle and reason correctly as to why it is twice as large.
We need as much as the first one to get something twice as big. Mona: You need to multiply everything by 2.	Their reasoning on how they constructed the rectangle is shallow, pursuing the idea that in order to make the figure twice the size, everything needs to be doubled. Their reasoning is furthermore not in line what they have actually done.
Teacher: Have you done that? Cia and Mona: Yes!	The teacher creates and asks a task-specific question to bring them to verify what they have done.
Teacher: Have you done this side twice as long? (pointing at the short side) Mona: No, that side needs to be the same length. So we did not change the amount of steps.	The students stick to their shallow reasoning and the teacher asks a prepared task-specific question, challenging the students to verify their reasoning.
Cia: . . . and not the corner, the cat needs to turn straight up even when we draw a rectangle, and straight is 90.	The students develop their reasoning and anchor their choices in intrinsic mathematics and refer to their programming, drawing the square and the rectangle, to do so.
Teacher: What will happen if you multiply all sides by 2?	The teacher creates and asks a task-specific question, addressing their former reasoning.
Cia: The rectangle will turn into a square. Mona: It will become too large. Teacher: How much?	The students develop their reasoning into what would have happened if they had done what they said they were going to do in the first place.
Cia: It will become 4 times larger.	

Summary: These students had initiated and seemed to have verified their reasoning as correct using Scratch. The teacher found that their reasoning – "We think it is twice as large" – was not anchored in mathematics and that their idea of doubling all values did not correspond to their programming. She asked task-specific questions to bring the students to *develop* and anchor their reasoning. The teacher chose task-specific questions to challenge Cia and Mona to realize and adjust their incorrect reasoning about doubling all values. The students eventually developed *creative mathematical reasoning* anchored in intrinsic mathematics that all straight turns are 90 degrees. These students furthermore made general assumptions that both squares and rectangles have 90-degree corners and that multiplying all sides in a square by 2 will make the square 4 times larger. These students moved from the *establishing practices mode* to enter the *critical discernment mode* using Scratch to verify their ideas. Finally, they entered the *situated discourse mode* using their programming (the square and the rectangle) to make general assumptions that straight turns are 90 degrees.

(C) ***Teacher-student interaction – Verifying reasoning***

In two of the situations concerning verifying reasoning the students needed task-specific questions to resume their problem-solving and engage in CMR. These students are trying to draw a triangle half the size of the square they constructed earlier (and have now erased). Their approach is based on the erroneous idea of taking half of all values compared to the square. However, they are not sure whether they succeeded and they ask the teacher.

Table C. Teacher-student interaction to verify reasoning

Dialogue	Analysis
Teacher: Well there's your triangle . . . but I can't see how big your square was . . . could you draw it again? (The students draw the square beside the triangle.)	The students will need both figures to compare areas, so the teacher creates and gives the students a task-specific suggestion, to redraw the square.
Teacher: Is the triangle half the size of the square?	Then she asks a prepared task-specific question prompting the students to find a way to test whether their idea is correct.
Anne: Yes . . . approximately. Teacher: I think you could draw it (the triangle) within the square. Lisa: Should we go the same way to here? (draws two sides of the triangle along two sides of the square) Teacher: Then you could tell if the triangle is half the size of the square. (leaves the students)	The students do not come up with a way to test their solution. The teacher creates and gives a task-specific suggestion on how to use Scratch to do so.
Anne: Ok, let's do it again.	

Summary: These students did not try to verify their idea, and when the teacher asked them to, they did not try to use Scratch to do so. From the original construction it was not possible to discern whether the triangle was exactly half the size of the square. The teacher used task-specific suggestions and questions to bring the students to test their idea using Scratch. However, she left it to the students to realize that the triangle was not half the size of the square. Lisa and Anne used Scratch to verify their reasoning as incorrect, and by doing so they moved from the *establishing practices* mode to enter the *critical discernment mode* using Scratch for verification.

(D) ***Teacher-student interaction, not supporting reasoning***

The following situation concerns a teacher-student interaction where one of the researchers, acting as a teacher, chose not to follow the idea of supporting students' reasoning. Instead, in order to help the students observe their mistake he showed a method to relate the programming to the figure the cat drew. The students had chosen two turning blocks to make the first turn. For a fairly long time they had been choosing different values (e.g., 90 degrees and 35 degrees) but without any reasoning supporting their choices. The cat turned too much, and they tried to correct their mistake by changing the value of the second turning block from 35 to 1. They realized that the cat turned too much and they gave up and asked one of the researchers for help.

Table D. Teacher-student interaction not supporting reasoning

Dialogue	Analysis
Cora: Our cat turns too much! Researcher: Let's see what's happening. (pointing at each block while explaining) The cat starts by putting down the pencil. Then the cat walks 180 steps. Then she turns 90 degrees. Then she turns once more . . .(Researcher stops)	The researcher has chosen not to apply the prepared interaction guidelines. Instead, he shares his reasoning, explaining what the program will do, step by step. He is silent when he comes to the block he interprets as their mistake.
Cora: No, the cat will not turn twice. [*Removes the "turn 1" block*] (The researcher leaves the students.)	One of the students removes the block and verifies that the program draws two perpendicular lines. The students do not develop their reasoning, however.

Cora and Maya have made a similar mistake when the cat makes the second turn; they have two "turn right" blocks (1 degree and 2 degrees). The cat does not turn enough, and they add a third turning block (15 degrees) without providing arguments for why they chose 15 degrees, and they try their program (Table 12).

Dialogue	Analysis
Cora: No, the cat did not turn enough. Maya: Let's try 50. Cora: No that did not work. Let's try 25.	Cora and Maya replicate their earlier strategy of changing values without any reasoning and then testing and failing. They have not tried to replicate the researcher's earlier reasoning, relating each of the blocks to the movements of the cat.
(Cora and Maya have tried to increase and decrease the value for a fairly long time but have not managed to make the cat turn straight.) They ask for help again.	

Summary: These students tried to verify their reasoning using Scratch. However, the cat did not move as intended and the students needed to develop their reasoning. The researcher did not try to elicit the students' reasoning by asking any general or task-specific questions. Instead, he tried to introduce a method to explore and draw conclusions from the programming blocks. Even though the students managed to change their programming to make the cat make a straight turn during the teacher-student interaction, they did not develop their reasoning. Hence, when making the second turn, they replicated the insufficient reasoning they had used when trying to make the first straight turn. The students did not engage in *formulating arguments*. They did not fully enter the *critical discernment mode* and were not able to discern how to adjust the program to construct straight turns.