



STEM education in the primary years to support mathematical thinking: using coding to identify mathematical structures and patterns

Jodie Miller¹

Accepted: 25 September 2019 / Published online: 1 October 2019
© FIZ Karlsruhe 2019

Abstract

Cross-curricula opportunities afforded by STEM education (Science, Technology, Engineering and Mathematics education), supports an environment where students can develop twenty-first century competencies. One approach to addressing cross-curricula opportunities in STEM education is the introduction of computer science (computer programming—coding) as a basic skill/literacy for all students. Coding (computer programming) is a language that draws on a set of syntax rules (or blocks for primary school students) that informs a computer program to execute a series of functions. While there is evidence that computational thinking (the thinking used for coding/computer programming) and conceptual development in mathematics are connected, there is limited research related to how such a confluence applies to primary school students. The aim of this article is to provide insight into how mathematical knowledge and thinking, specifically the identification of mathematical patterns and structures, can be promoted through engagement with coding activities. The data for this article is drawn from year 2 students ($n = 135$) in two Australian primary schools. A teaching experiment approach was adopted for the study with a small intervention group ($n = 40$) undertaking coding lessons for 6 weeks. Data collection comprised of pre-test and post-tests with a focus on patterning and coding in conjunction with video-recorded lessons. The study provides evidence that the learning that takes place through coding instruction can lead to higher levels of students' mathematical thinking in relation to identifying mathematical patterns and structures that can lead to generalisations.

Keywords Mathematical thinking · Primary · Patterning · Coding

1 STEM, computational thinking and mathematics education

Twenty-first century competencies/skills are constructs referred to in international education policy that seek to address current and future global workforce requirements through promoting the development of cognitively demanding learning, and inter- and intra-personal skills for learners (Voogt and Roblin 2012). While there is variance between each policy and strategy, twenty-first century competencies promote learning and innovation skills, information, technology and media skills, and life and career skills (Partnership for 21st century skills, 2019). It has been argued that the cross-curricula opportunities

afforded by STEM education (Science, Technology, Engineering and Mathematics education), provides the best environment for students to develop these competencies. For example, students who are equipped with STEM knowledge are capable of identifying, applying and integrating the underpinning concepts and thinking skills to solve complex problems and generate innovative solutions (Meng et al. 2013) in a technologically driven world. One approach to addressing this in the curriculum is the introduction of computer science (computer programming—coding) and computational thinking as a basic skill/literacy for all students (García-Peñalvo et al. 2016; Prensky 2008). It has been argued that children who have a strong foundation in computational thinking develop twenty-first century competencies in particular, children are found to be more effective problem solvers and critical thinkers (Wing 2006). The focus of this paper is to examine the twenty-first century skill of computational thinking, that is to recognise patterns, decompose, abstract and create

✉ Jodie Miller
jodie.miller@uq.edu.au

¹ University of Queensland, St Lucia, Brisbane, Australia

algorithms in computer programming contexts to solve (mathematics) problems (Papert 1980).

Despite it being widely acknowledged that mathematics and computational thinking (the thinking used for coding/computer programming) are fundamental to all STEM disciplines and critical for twenty-first century learners (Prinsley and Johnston 2015), there is limited evidence to inform how primary school students develop the mathematical thinking using STEM technologies such as coding. In response to the identified challenges in STEM education, international curriculum reform has emphasised the implementation of coding in primary school (e.g., Australian National Curriculum; Queensland Advancement Plan; Canadian STEM Agenda). In Australia, this is reflected in the new Digital Technologies (DT) curriculum, which emphasises the development of higher order computational, problem solving and creative thinking skills, including a deep engagement with coding (ACARA 2018). While previous research has established a connection between teaching and learning coding and the development of general mathematical capability (e.g., Benton et al. 2017; Miller and Larkin 2017), there is limited research on how primary school students develop mathematical thinking, in particular recognising patterns, decomposing, abstracting and creating algorithms; an essential component of computational thinking in coding contexts.

Despite the noted benefits for integrating STEM into the school curriculum, that is capitalising on opportunities to make connections between the four disciplines of science, technology, engineering and mathematics rather than teaching them in silos, international research in this area is in its infancy and raises many concerns for teaching and learning practices (English 2016). Effectively integrating STEM and overcoming curriculum obstacles are some of the challenges facing national and international researchers and educators. Of concern, is the lack of a cohesive understanding with respect to what an integrated STEM curriculum involves, coupled with inadequate teacher knowledge of multidisciplinary STEM content areas (Moore et al. 2014). Within the Australian Curriculum, coding has been aligned with the digital technologies area (ACARA 2018), however, no model or framework is provided for coding to demonstrate how this can be integrated effectively into mathematics or other curriculum areas. Currently, coding is taught only in technology lessons, with a disciplinary focus rather than adopting an interdisciplinary approach, which draws on science, mathematics and engineering (Vasquez et al. 2013). Of concern, is the lack of explicit links to mathematics in new digital technologies curriculum. Hence, this study seeks to identify connections between coding within the STEM curriculum underpinned by mathematics.

1.1 Literature

The following sections address the literature with regards to the importance of patterning and structure in relation to mathematics, then an examination of the literature in relation to mathematics and computer programming. Following this, the research questions will be presented.

1.2 Mathematics, patterning and structure

Mathematics is founded on patterns, and thus, engaging and experiencing patterning is essential from an early age. A mathematical pattern is described as ‘any predictable regularity, usually involving spatial, numerical or logical relationships’ (Mulligan and Mitchelmore 2009, p. 34). Experiences such as sorting and classifying activities are often the initial experiences students undertake when beginning to investigate patterns. Following this, students participate in activities that involve kinaesthetic movement, concrete manipulatives, space, pictorial representations and numbers as they copy, continue, complete and create repeating patterns (Warren et al. 2012). A repeating pattern can be defined as a pattern in which there is a discernible unit of repeat. This is a cyclical structure that can be generated by the repeated application of a smaller portion, or discernible units, of the pattern (Zazkis and Lijedahl 2002). These patterns can range in levels of complexity. Commonly, a repeating pattern that has an AB discernible unit of repeat is introduced to students (e.g., ABABABABA; jump, clap, jump, clap; night, day, night, day). Research has indicated that it is important for students to be able to identify the unit of repeat as it is necessary for later mathematical development and concepts such as generalisation (Zazkis and Lijedahl 2002). Thus, from a conceptual point of a view, it appears as if it is much more important to identify the repeating unit (the structure of the pattern), than it is to be able to create complex repeating patterns. However, past research indicates that identifying the unit of repeat are more challenging for young students to grasp (Lüken 2018; Rittle-Johnson et al., 2013; Threlfall, 1999).

Patterning forms the basis for young students to recognising mathematical structures and engage with early algebraic thinking (Blanton and Kaput 2011; Cooper and Warren 2011; Miller and Warren 2012). In addition, it also compliments other mathematical concepts such as counting, multiplicative thinking, arithmetic structure, and measurement (Cooper and Warren 2011; Papic, Mulligan, & Mitchelmore 2011; Warren and Cooper 2007). Importantly, a students’ capability to pattern in primary schooling has been proven to impact on students’ mathematical

achievement (Papic 2007; Warren and Miller 2013). It appears to be essential that young students develop a robust understanding of patterning during primary school years. Research highlights that young students can successfully engage in a range of patterning tasks including experiences in repeating patterns, growing patterns and functional thinking which leads to early algebraic thinking (Blanton and Kaput 2011; Cooper and Warren 2008). The use of patterns in the early years provides students with the opportunity to apply rules, reason and move to abstract notations in mathematics. However, fundamental to this is the aptitude for young students to recognise structure in patterning and mathematics. The identification of the unit of repeat and the translation of the pattern into other modes or representations allows students to build an understanding of the structure within the pattern itself.

1.3 Mathematics and computer coding

The teaching and learning of mathematics through coding (e.g., Clements et al. 2001) and programmable robots (e.g., Highfield 2015), have indicated that both of these STEM areas promote mathematical learning including problem solving, measurement, geometry and spatial concepts (Savard and Highfield 2015). Furthermore, quantitative studies have determined that there is a correlation between coding using the programming software Scratch (a child friendly blocks-based programming language developed by MIT) and mathematics test scores for year 4 students (Lewis and Shah 2012). Past research indicates that coding provides an opportunity for developing students' mathematical knowledge and cognition (Papert 1980). In particular, it is acknowledged that computer programming offers a platform for students to apply algebraic thinking (recognising patterns, generalising structures in a context outside of mathematics classrooms. Much of the research in this area has focused on students in upper primary contexts (Hoyles 1985), or has only examined how students use algebraic variables (Noss 1986); or how teachers develop mathematical content knowledge in relation to algebraic variables while teaching computer programming (Clark-Wilson and Hoyles 2017). However, despite three decades of research in the area of coding and mathematics (e.g., Benton et al. 2017; Hoyles and Noss 1992), there are limited studies in the area of coding and developing understanding in mathematical patterning and structures; the key underpinning concepts for later algebraic thinking. This study contributes to a new knowledge in this field by examining young primary school students' development of patterning (repeating and growing patterns), structures (recognising underlying structures) and mathematical thinking (generalising), while using a newly developed coding program (Scratch). This moves beyond

the previous studies which have focused on upper primary students developing an understanding of algebraic variables.

The aim of this article is to provide insight into how mathematical knowledge, specifically the identification of mathematical patterns and structures, can be promoted through engagement with the means most often used to implement computational thinking in school classrooms—coding. The research questions that are examined in this paper are:

- What were the changes in year 2 primary students understanding of mathematical patterns and structure after undertaking six coding lessons?
- What types of concepts of mathematical thinking (patterns and structures) do year 2 primary students display while undertaking coding activities?

1.4 Theoretical Framework

This research is underpinned by constructionism as defined by Papert and Harel (1991). From this theoretical standpoint learning is 'building knowledge structures through progressive internalization of actions...' (Papert and Harel 1991, p. 1), which often takes place in situations where students are learning through doing. For example, in this study students are constructing and internalising their knowledge of mathematical concepts as they engage with digital learning experiences (external aids). As learners engage with the external artefact, they converse with themselves, or others, and construct and reconstruct their knowledge through 'doing'. Thus, building knowledge as a part of experiences and social interactions (Stahl, 2003). This theoretical approach to learning assists researchers to build an understanding of how students construct or deepen their knowledge and how this is expressed through different external artefacts or representations with others. Ackerman emphasises that the 'emphasis shifts from universals to individual learners' conversation with their own favourite representations, artefacts, or objects-to-think with (Ackerman 2001, p. 4). Importantly, constructionism identifies that the learner is active in the construction of knowledge through the use of external artefacts that are shared by learners. This moves beyond the idea of merely producing knowledge with an external artefact, and emphasises the importance of sharing the representations of understanding or producing their knowledge within social interactions with others (Han and Bhattacharya, 2001).

Constructionism is an appropriate theoretical framework for this study as it allows the researcher to explore how students construct concepts of mathematical thinking, examine how this is displayed as they work with coding programs (external artefacts), and how they articulate their understanding with others. This theoretical framework informs the design of the research including the selection of tasks. For

example, all students work and represent their knowledge and understanding through the use of an external artefact and in this case that is the Scratch computer program. In addition, it also informs the ways in which students work through the tasks. For example, the students initially solve the problem as an individual, trialing and re-trialing their computer code constructing their understanding. At the same time this is providing the students with opportunity to build their own knowledge structures of mathematics. Students then share their coding ideas using the representations of their code with their peers to refine or reconstruct their understanding of the problem, further deepening their knowledge. It emphasises the importance of the student learning though doing while engaged with the external artefact (coding program—Scratch), using a STEM tool as an expressive medium.

2 Methodology

2.1 Participants

One hundred and thirty-five year 2 students (aged 7–8 years old) from two schools located close to a major city in Australia participated in the study (School A—63 students; School B—72 students). Gender was balanced with 70 girls and 65 boys. Both schools were considered to be of average socioeconomic demographic, with students representing a diverse population including both Indigenous students (Aboriginal and Torres Strait Islander students) and students who identify as having English as an additional language. Parents provided written permission for their child to participate in the study. Following pre-testing a smaller sample of students ($n=40$) were selected to participate in six coding lessons. These 40 students were across both School A ($n=16$) and School B ($n=24$).

An a priori statistical power analysis was performed to determine the sample size estimation for the intervention group. There was no prior pilot study data to draw on. As such, the effect size in this study was determined to be medium (0.5) using Cohen's (1988) criteria. With an $\alpha=0.05$ and a power 0.8. The projected sample size needed to determine effect was approximately $N=34$ for a paired t test measure. Thus, the proposed sample size of $N=40$ would allow for any attrition and other factors impacting on the study.

Selection of students for the intervention was based on their demonstrated prior knowledge of patterning and coding from the pre-test data. The students were to represent a typical classroom with a diverse range of learners (low, medium, and high achievers). As such, there were four identified groups from which the students were randomly selected from: low patterning/low coding; low patterning/

mid coding; mid patterning/low coding; and mid patterning/mid coding. From the analysis of the pre-test data there were no students who demonstrated to have both high test scores in patterning and coding. Each of the four subgroups consisted of 10 students with an even number of male and female students in each.

2.2 Research design

This study employs a teaching experiment methodology (Confrey and Lachance 2000), using both quantitative and qualitative methods, to develop an understanding of the types of mathematical thinking year 2 students demonstrate as they engage in coding lessons. A teaching experiment presents a living methodology where students' mathematical learning and transformations of knowledge can be explored within their own classroom environment (Steffe and Thompson 2000). It builds on the constructionism theory that believes students are able to construct mathematics (Papert 1980). With this, it is essential to include arguments for how mathematics is constructed from diverse constituents. This study utilised a teaching experiment for the primary purpose of building an understanding as to how students construct their mathematical knowledge (Steffe and Thompson 2000), specifically the identification of mathematical patterns and structures that can be promoted through engagement with coding lessons. The teaching experiment comprised of: (i) pre-testing; (ii) 6×45 -min lessons of coding lessons (one lesson per week for 6 weeks); and (iii) post-testing.

The aim of the teaching experiment was to explore how students developed mathematical knowledge and thinking, in particular patterns and structures, as they participated in coding lessons. As a part of the intervention the researcher and research assistant in consultation with the classroom teacher, assumed the role of the teacher during the coding lessons with the smaller intervention group. This was because the teachers had little knowledge as to how to teach coding or make delineations to mathematics during these lessons. In addition, in order to have an intervention group it was essential that the researcher could undertake these lessons so that the classroom teacher could remain with his/her class.

The role of researcher as teacher was a key part of the teaching experiment. The researcher observed students as they participated in learning experiences. The role throughout the lessons were to: (a) support students to articulate their understanding of the code they had created; and, (b) encourage students to justify their choices to their peers making connections to mathematics. This was achieved by:

- asking students questions at pertinent times in the activity. This was done on an individual basis as well as a whole class;

- encouraging students to trial their code, identify errors and make corrections;
- have students share and justify their coding to their peers;
- asking probing questions that encouraged students to attend to the mathematics in their codes in particular to consider the patterns and potential generalisations; and,
- facilitate class discussions regarding a range of approaches to the coding process.

2.2.1 Pre and post testing of patterning and coding

At the commencement of the teaching experiment all students participated in pre-testing measures to identify students prior patterning knowledge and coding knowledge. As it was assumed coding and mathematical patterning were related, it was decided to test the students on these two constructs. The tests were developed from prior research on patterning and structures (e.g., Cooper and Warren 2008). Items included students identifying repeating patterns, making predictions about patterns, generalising patterns, and creating patterns. In total there were 10 items on patterning, with a total possible score of 30. Test items were then developed in coding contexts; each of the patterning and structure concepts were mapped onto coding contexts, there were 10 items on coding with a total possible score of 10. Figure 1 provides examples of the test items, in particular identifying the repeating part of the pattern/code.

To ensure consistency in the administration of the test, the researcher administered all tests across both school settings. In addition, each test was accompanied by a script that provided the information the researcher and research assistant said as they read the instructions for the test as well as the question for each test item. The test was administered by the

researcher to the whole class in a single session. Administration of the pre-test was within the first couple of weeks of Term 1 commencing (February). There was approximately an 8-week period between the administration of the pre and post-test (late April).


2.2.2 The intervention

The teaching experiment consisted of six lessons, three with a coding focus using Scratch and three coding robots. Each lesson focused on teaching a mathematical concept using coding or robotics (e.g., drawing a square, drawing spirals, moving a robot through a particular path). The intervention was the major data gathering method in this study. These lessons focused on identifying how students develop mathematical thinking while coding. All lessons were conducted during school time, outside of the students' classroom. While the selected students were participating in the intervention for the 6-weeks (6 × 45-min lessons—one lesson each week for six weeks), the year 2 teachers refrained from teaching robotics and coding for the 6 weeks. While the small cohort of students participated in the intervention, the control group stayed with their classroom teacher and participated in normal class lessons (e.g., English or mathematics) as planned by their teacher for that time. On all other days where the intervention was not occurring all students participated in mathematics lessons. Prior to the study, students had no exposure to coding or robotics in these schools. This ensured a control group with for the study. The small selected cohort of students attend all six lessons over the 6-week period. The researcher conducted the lessons for the intervention, as they did not form part of the formal teaching

8. Coding

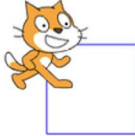

I have used this code to draw a square.

3a. Circle the repeating part



3b. What shape will be at position 13?

3c. What shape and colour will be at position 25?

a. Circle the repeating part of the code?

Fig. 1 Identifying repeating pattern test items on the patterning (Q3a) and coding test (Q8a)

in the classroom. The lessons were conducted in a classroom where students had access to laptops and robots.

Each lesson in the intervention, commenced with a series of research-developed open-ended tasks (Sullivan 2011) employing coding that require mathematical thinking. Tasks ranged in difficulty and mathematical concepts (e.g., geometry, measurement). Students first, attempted the task individually and then worked in pairs sharing and discussing their codes to solve the problem presented. The decision for this was to begin to see how students were initially thinking about coding their tasks and what links they were making (if any) to mathematics. It was essential to see how students were constructing their understanding and what issues may potentially arise from the designed tasks. As the students worked through the tasks, the researcher asked questions to probe student understanding. Examples of these questions could include: How did you code your Scratch cat/robot to rotate 90°? How many rotations were needed to move the Scratch cat/robot to that destination? What mathematics did you need to use to solve this problem? After the students created their initial code they then shared their code with other class members either in groups of two or three to share their different strategies and troubleshoot (debug) issues in their code. It was at this point students shared different ways of coding and assisted students who were having difficulties. At the end of the task, the whole class came together to share their ideas.

Two video cameras were used to collect data during each lesson of each teaching experiment, with one camera focussed on the researcher and one on the group of students. In addition, the researcher and research assistant had iPads which they used to record closer conversations between pairs of students and also photograph students' codes on the computer screen. These video-recordings and photographs were used for in-depth analysis by the researchers. In this paper, only the findings from lesson one and five from the teaching experiment are presented, where students were required to use the Scratch program to draw a square and a spiroilateral. There are no data presented with respect to students using robotics in this paper.

2.3 Data analysis

Quantitative Data Analysis The analysis of the pre-post testing measured the changes in students' understanding of mathematical patterning and seeing patterns and structures in codes before and after the intervention. In addition, the pre-post test scores were used to compare with those students who did not participate in the intervention (control group vs intervention group). Data analysis included, frequency scores for test items, paired t-tests and eta² scores. Students responses for the pre- and post-test questions were

input into a spreadsheet and then automatically marked as either correct (1) or incorrect (0). The total possible score was 40 (combined score—30 patterning and 10 coding). Analyses were performed on the raw data to ensure the accuracy of the data entry and eliminate any that may have occurred. Data were then transferred to a statistical package (SPSS) where further statistical analysis was conducted such as paired t tests and eta² scores.

Qualitative Data Analysis The intervention required in-depth analysis using an iterative approach (Lesh and Lehrer 2000). The videotape was analysed using iterative refinement cycles to determine student conceptual change (Lesh and Lehrer 2000). First, the initial video-footage of the intervention were transcribed to capture students' verbal responses. Analysis of these transcriptions were undertaken to consider emerging key mathematical themes from the intervention (e.g., apparent mathematical concepts). Second, the data was analysed focusing on types of mathematical thinking students demonstrated as they used coding. A constant comparative method of analysis was used as the systematic approach analyse the lesson data. There were three iterative coding procedures undertaken: open coding, selective coding and axial coding. Open coding of the data provided the opportunity for the researchers to examine the initial transcript data to identify similarities and differences of students mathematical thinking to establish initial categories (Creswell 2008). Axial coding was undertaken to examine the established codes and identify the connectedness between categories and the existing theories. Finally, selective coding was employed to examine the interrelationships between the codes to determine a deeper understanding of the research and potential theories that emerged (Creswell 2008). Checking reliability and consistency of the codes between the researcher and research assistant were conducted at each cycle to verify the coding of the transcripts.

3 Results

This section reports on the analysis of the pre and post tests and provides case examples from the teaching experiment that demonstrates the types of mathematical thinking students engaged with during the lessons.

3.1 Results of the pre and post-testing

Research Question 1: What were the changes in year 2 primary students understanding of mathematical patterns and structure after undertaking six coding lessons?

Analysis of the pre and post test data focusing on students understanding of patterning and coding combine (Table 1), the patterning test (Table 2) and the coding test (Table 3) are

Table 1 Pre and post test score analysis for control and intervention student groups for the patterning and coding tests combine

	Pre-test mean (sd)	Post-test mean (sd)	Gain score	t	d	p
Control Group (n=95)	6.27 (3.73)	13.07 (6.29)	6.80	13.65	1.34	0.001*
Intervention Group (n=40)	9.05 (4.99)	20.08 (7.25)	11.03	13.96	1.77	0.001*

*Statistically significant

Table 2 Pre and post test scores analysis for control and intervention student groups for the patterning test

	Pre-test mean (sd)	Post-test mean (sd)	Gain score	t	d	p
Control group (n=95)	5.81 (3.33)	11.95 (9.68)	6.15	13.46	0.84	0.001*
Intervention group (n=40)	8.42 (4.51)	16.30 (5.48)	7.88	11.48	1.57	0.001*

Table 3 Pre and post test scores analysis for control and intervention student groups for the coding test

	Pre-test mean (sd)	Post-test mean (sd)	Gain score	t	d	p
Control group (n=95)	0.46 (0.74)	1.11 (1.12)	0.64	5.59	0.68	0.001*
Intervention group (n=40)	0.62 (0.77)	3.78 (2.35)	3.15	9.74	1.80	0.001*

presented in tables below. In particular this data represents how students were recognising and deducing patterns. Data reveals that the teachers were not teaching coding during the 6-week intervention during class time as the students' scores from the control group remained relatively static.

Both groups had a significant increase between pre- and post-testing mean scores with regards to development in mathematical patterning and coding with the test scores combined (Table 1). To further identify where the gains were made it is evident that both groups improved on the patterning test, with the intervention students performing better than the control group. In addition, the intervention group were able to identifying patterns and structures on the coding test while the control group remained static. Analysis of the effect size (d) indicated that the coding intervention had a large effect on all three test scores (> 0.50) (Cohen 1988). As a means to understand what this effect size means in relation to teaching and learning, Hattie (2009) reported that an effect size of 0.4 is average for all studies and that teachers typically attain this in a school year. Anything above this can be as a result of a program or intervention implemented in the school and is labeled the zone of desired effect. Thus, the reported effect size of this study indicates that not only are the results statistically significant but also potentially educationally significant.

Research Question 2: What concepts of mathematical thinking did year 2 primary students display while undertaking coding activities?

To address the following research question two lessons are presented as case studies to demonstrate the types of mathematical thinking in relation to patterning and mathematical structures that students demonstrated while undertaking coding lessons. The first case presented was the initial

lesson in the intervention. Students were presented with the problem to draw a square using the Scratch programming software. The second case presented is taken from a coding task that focused on moving the Scratch cat to draw a spiro-lateral. This lesson was the fifth lesson undertaken as part of the intervention. The reason these two cases are selected were because these are two lessons that involved students using the Scratch program. The problems for these lessons were posed to students and they initially had the opportunity to solve these independently before working more collaboratively. The third lesson is an open-ended task where students worked in pairs to solve a problem. The additional three lessons, focused on students transferring their coding knowledge to a robotics context. Thus, these two lessons give the best insights into how young students develop concepts of mathematical thinking while undertaking coding activities.

Case lesson 1: Drawing a Square on Scratch

This is the first lesson that was undertaken as part of the intervention. Forty students participated in the 45-min lesson that used the computer coding program Scratch. At the beginning of the lesson the students were asked to share with the class the features of a square. The following indicate the types of responses students provided:

Sarah: A square has 4 straight sides.


Tom: All sides need to be the same length.

Charlie: It has four corners.

Following, this the students were posed a problem—Can you help Scratch cat draw a square? Table 4 displays the tasks with the anticipated mathematical concepts and coding concepts students may employ to the tasks.

Students were shown some basic features by the researcher, including the types of blocks and how to drop

Table 4 Drawing a square task with the anticipated mathematical and coding concepts

Task	Mathematical concepts	Scratch coding concepts
Scratch Cat's favourite shape is a square. Can you help Scratch Cat draw a square? 	Discuss properties of a square Testing and measuring lengths. How far is one step in Scratch? Testing and measuring Angles/turns Identify any repeating patterns	Explore the coding blocks and symbols used in scratch Measuring in pixels Building a code Running a code Editing a code if there are errors Using the repeat/loop function

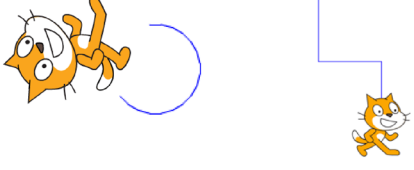
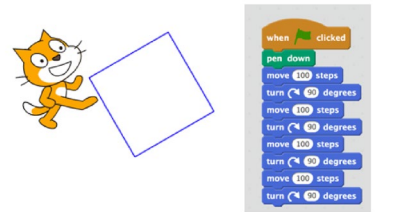
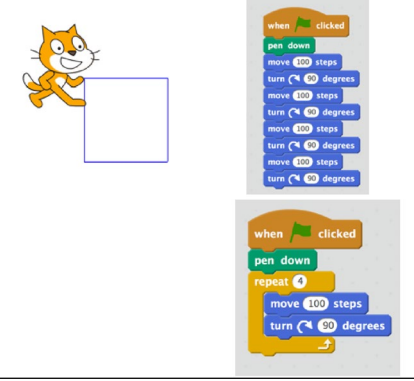
and drag the blocks of code. Students then worked initially independently to solve the problem. Once they had created their code they shared it with their peers.

The video-data were analysed to elicit the types of mathematical thinking displayed by students. With a focus on patterning and structure, from the analysis there were three themes that emerged from the “Draw a Square” Task. This was the students first attempt to draw a square. Table 5 displays the student approaches, the frequency of

the types of responses, and an example of a student’s work fitting this type.

Data revealed that 28 students could draw a square using the Scratch program. In particular, students were able to identify the repeating part of the pattern. In addition, the responses demonstrate that the year 2 students were at a higher level than what was required by the year 2 Australian mathematics curriculum (ACARA 2018). For example, in Australia, students in year 2 do not need to know how to measure turns using

Table 5 Student response and explanation of approach, frequency of student responses, and example of student’s work

Response and Explanation	Frequency	Example
<i>I can't draw a square.</i> Student attempted to draw a square but used 15 degree turns. Students did not construct a square as they alternated the turns to the right and left.	12	
<i>Is this still a square?</i> Students correctly coded a square but were unsure whether it was a square or not because of the orientation.	3	
<i>I made a square.</i> Students were able to write a code to make the Scratch cat draw a square parallel to the bottom of the screen. Five students identified that they can see a repeating pattern and used the repeat coding block to draw a square.	25	

degrees. There were three areas in particular that demonstrated the higher levels of mathematical thinking: (i) converting their knowledge of quarter turns to work with metric measures of 90° turns, (ii) orientation and perspective taking, and, (iii) deducing a repeating pattern to provide a generalised code for making a square.

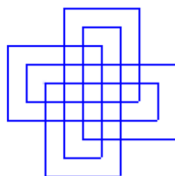
After their initial trial, the group of students came back together and discussed their different coding attempts. Following the class discussion facilitated by the researcher, many students who were unable to draw a square in their initial attempts were able to find the error in their code and problem-solve to amend the code so the Scratch cat could draw a square. This was a result of students working cooperatively to assist one another to fix their codes. Additionally, the researcher asked students probing questions to focus on where they saw the error that occurred between the drawing and the block code language. The students were able to make connections between the mathematical error and the coding language. Once amended, these students were then successful coding the squares.

Students then worked in pairs to find a generalised code for drawing the perimeter of the square. An example provided by a student in the whole class discussion is presented below:

Jesse: If you wanted to draw any square you just do the length and a 90 degree turn, four times. Then to work out the length of the outside of the square it is just any length you put in like 30, 40 or 100 and then times that by four. So, a square with a side length of 40 would equal 160 steps in total.

It was evident in the lesson that students were making connections between their codes and the generalised structure of perimeter. They were able to identify their own generalised rule for the perimeter of a square. It appears that through the use of Scratch students were able to explore patterning—in particular repeating patterns, and then use this code to develop their own generalised rules for perimeter. Unlike a typical year 2 classroom where students usually draw squares (using paper, pencil and ruler) and identify that squares have four straight sides that are all equal, these students were then able to make further connections. For example, students were also able to articulate squares have interior angles of 90° . It can be assumed that this technology provided a platform for students to see the mathematics in a deeper way and build a more connected understanding.

Fig. 2 Spirolateral shown to students



Case lesson 2: Drawing a Spirolateral

At the beginning of the lesson, the researcher presented an image of a Spirolateral to the students (see Fig. 2). Students were asked to think of possible codes that would assist us to draw this shape. Figure 2 presents the figure shown to students.

Students offered their insights as a whole group before undertaking the task. The researcher asked the students, “What do you notice about this drawing?” One student identified and shared with the group the following features about the original figure (see Fig. 3):

Archie: It looks like there are four parts to the drawing (Archie moves to the board and uses his finger to draw circles around the four parts he sees). It looks like there are rectangles inside of rectangles.

Then students were shown just one section of the spirolateral to prompt them into the task (see Fig. 4). The researcher asked, “When we focus on one of these sections, what do you notice?” Students indicated that they could see that there were sections of the spirolateral that was repeating and growing. One student articulated that:

Bree: It moves one length, maybe 100, and then turns 90 degrees, then moves the same length, then turns 90 degrees, then it gets longer. Maybe it doubles the length. Then it repeats it again.

The students worked individually to draw a plan of the spirolateral before undertaking coding. Here is an example of one plan a student drew and then the code they developed in Scratch to draw this section of the spirolateral (see first image in Fig. 5). Finally the students were able to articulate that they needed to repeat this pattern four times to draw the complete spirolateral. This is depicted in the last image in Fig. 5 where the repeat loop is wrapped around the code.

Following students planning and coding the spirolateral, a class discussion was held. During the discussion students made the following observations.

Fig. 3 Students’ identified structure of the spirolateral

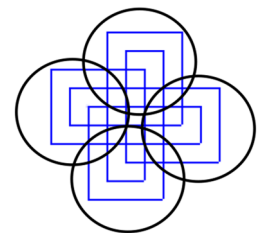
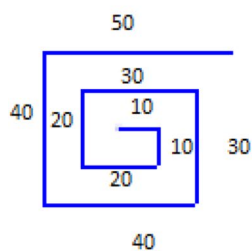


Fig. 4 Section of spirolateral shown to the students



Fig. 5 Student planning and coding developed in Scratch



Olive: I can see the pattern is both growing and repeating itself

Michael: Yeah, it is growing the length each time. Oh, I mean, you do it twice then it goes up in ten.

Daniel: I can also see it going up in move ten steps, move 20 steps, move 30 steps. All the way to 50.

Researcher: If you needed to explain this code to your friend what would you tell them?

Saige: You move a number of steps and then turn 90 degrees—two times, then you go move 10 more and turn another—two times. This keeps going. You do the same thing two times then add on 10 to the moves. It is like a repeating pattern and then you do a big repeating pattern 4 times when you have finished the first spiral part.

Daniel: But, you don't do the last one two times, because it needs to move on to make the new spiral.

Olive: It is both a growing and a repeating pattern.

It can be seen from the transcript above that students were able to identify the structure of the spiroilateral and transfer this to the computer code. Interestingly, they were able to articulate that they could see growing and repeating pattern structures in the code. Some students were also able to describe some generalisable features of the pattern. It appears that these types of tasks, provided the opportunity for young students to think in more abstract ways about mathematics. It is conjectured that the use of the technology in conjunction with the mathematical tasks supported this to occur.

4 Discussion

The results of this study indicate the potential of a coding intervention on providing a platform for students to see mathematical structures and patterns in codes. In particular, developing the ability to see patterns, recognised the unit of repeat, deduce the pattern and abstract the general structure; key components of computational thinking as a twenty-first century skill. The results of the pre- and post-tests demonstrate that students who participated in the intervention had significant growth in their understanding of, and recognition of, mathematical patterns and structures in coding contexts. These findings begin to address and provide insight into the three decades of research in coding and mathematics (e.g., Benton et al. 2017; Hoyles and Noss 1992), that have yet to focus on the benefits of developing this area of mathematical thinking, particularly with young students. Furthermore, the findings from this study address the potential impact coding has on students' mathematics acquisition (Benton et al. 2017). It appears that an intervention that focuses on patterns and structures in coding contexts improves both students' mathematical understanding of patterns and its application in coding contexts. Students were able to self-generate generalisable rules for perimeter of a square drawing while engaging with coding; a real world application of mathematics. In addition, the findings from this study add to the current body of research, and demonstrates that patterning appears to be an important underlying skill, or STEM practice, for students to develop a deep understanding and

application of mathematics in these contexts. With regards to research relating to patterns, structure and computational thinking the results of this study support the conjecture that there is a relationship between one's ability to code and their understanding of mathematical patterns.

Students who participated in the intervention were able to identify repeating and growing patterns within coding applications. The ability to discern patterns, identifying the unit of repeat, is the precursor to generalising mathematics (Threlfall 1999), and generalising a pattern is central to computational thinking. It was evident that in seeking to reduce codes students needed to find the unit of repeat and then wrap this in the looping function. The loop (repeat function) function gave way to this, and reduced the amount of code students needed to produce, as evident in the spiroilateral lesson. It is impractical for students to produce lengths of code, and thus, they are pushed to generalise or notice the unit of repeat to reduce the amount of coding required to draw a mathematical shape on the screen. This demonstrates there is a potential link between recognising units of repeat in coding contexts and then transferring this knowledge into mathematical contexts. This skill links to that of computational thinking, and provides some insight into the stepping stones as to how students can perform abstractions when working with technology to explore mathematics from a young age.

Typically, instruction in primary school mathematics overly focuses on arithmetic procedures at the detriment of students developing the knowledge and skills of mathematical thinking, in particular identifying patterns, extracting structures and forming generalisations—key concepts in early algebra (Cooper and Warren, 2011). It is evident that students were beginning to engage with early algebraic thinking including forming generalisations when writing computer codes. Teaching mathematics through coding may provide opportunities for students to engage in applied mathematical thinking that differs from traditional teaching foci (Miller and Larkin 2017). Thus, coding has the potential to provide a platform for a higher and more connected engagement in mathematics than what normally occurs in most primary school classrooms. There is an opportunity to capitalise on the integration of this technology in mathematics classrooms to support students to engage with this type of thinking that supports the development of twenty-first century skills, particularly computational thinking. It is with this development of deeper understanding of mathematical structures that students may establish stronger foundational STEM knowledge. This study begins to demonstrate the effectiveness of a novel pedagogical approach aimed at promoting young primary school students' mathematical thinking via engagement with coding activities; generating new knowledge about students' capacity to engage with deeper levels of mathematics connected and applied in real world contexts (coding).

It can be assumed that this type of intervention supports students to develop a more connected understanding of STEM, in particular the way in which mathematics is embedded in technology. This study begins to address the limited evidence to inform, how mathematics and computational thinking are fundamental to STEM disciplines and critical for twenty-first century learners (Prinsley and Johnston 2015). It is evidenced that primary school students can develop mathematical thinking using STEM technologies such as coding. This has implication for international curriculum and international policy that has emphasised coding in primary school contexts. It highlights the importance of an integrated approach to teaching technology and mathematics; bridging the two fields. Findings from this research suggests that there is a need for a more connected curriculum that moves beyond the separation of technologies and mathematics to integrate STEM and promote thinking that is embedded and applied across the curriculum areas.

5 Concluding remark

The aim of this article was to provide insight into how mathematical knowledge and thinking, specifically the identification of mathematical patterns and structures (recognising patterns, decomposing, abstracting and creating algorithms) can be promoted through engagement with coding lessons. The design-based intervention demonstrated that a 6-week program focusing on coding has a significant effect on students' capability to identify and generalise patterns and structures, more so than students who were not participating. Thus, it is conjectured that computer coding offers primary school students a new means for exploring, applying and promoting patterns and structures that has the potential to lead to other mathematical concepts such as early algebraic thinking—in particular the ability to generalise mathematical structures. In addition, it appears cross-curricula opportunities afforded by STEM education (bridging Technology and Mathematics education), provides an optimum environment for students to develop these twenty-first century competencies (e.g., computational thinking; abstract thinking). However, it is acknowledged that there is a need to upskill teachers to 'see' the mathematics in coding lessons as this is not always apparent in the curriculum. As such, school systems need to provide opportunity to upskill and support teachers in the implementation of new technologies and skills such as computational thinking.

Furthermore, as the study is bound by context and time, it is acknowledged that there are limitations to this research and there is a need to undertake the study with a larger sample of students with prolonged observations. In addition, by having a larger sample size the test constructs of both patterning and coding can be separated to determine the

interactions between them on students' capability to code and pattern. Undertaking further research may provide the opportunity to determine, which of these two constructs influences the other, for example if you are better at patterning are you more likely to be better at coding/computational thinking? Following this initial study and preliminary findings, it is essential to undertake a larger project to examine the effects of this type of intervention on early algebraic thinking and mathematics more broadly. In particular, a larger study to build an understanding of how young primary school students can develop algebraic thinking concepts (e.g., functional thinking; generalisations) through coding contexts. Through building a deeper understanding of the connections between mathematics and twenty-first century skills such as computational thinking, educators will be able to provide a more authentic and applied approach to mathematics lessons for primary school students, promoting an interest in STEM.

Acknowledgements I would like to thank Emeritus Professor Elizabeth Warren for her continued mentorship and feedback for this study. I would also like to acknowledge Angela Hennessey for her work as the research assistant on this project, as well as the schools and students who have participated in the study. This research first appears in a MERGA conference paper (Miller and Larkin 2017).

References

- Ackerman, E. (2001). Piaget's constructivism, Papert's constructionism: What's the difference? http://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf. Retrieved October 24, 2018.
- Australian Curriculum, Assessment and Reporting Authority [ACARA]. (2018). *The Australian curriculum*. <https://www.australiancurriculum.edu.au/>. Retrieved May 16, 2018.
- Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, 3(2), 115–138.
- Blanton, M., & Kaput, J. (2011). Functional thinking as a route into algebra in the elementary grades. In J. Cai & E. Knuth (Eds.), *Early algebraization: A global dialogue from multiple perspectives* (pp. 5–23). Berlin: Springer.
- Clark-Wilson, A., & Hoyles, C. (2017). *Dynamic digital technologies for dynamic mathematics*. <https://www.nuffieldfoundation.org/sites/default/files/files/Hoyles%2041909%20-%20Executive%20Summary.pdf>. Retrieved January 12, 2018.
- Clements, D. H., Battista, M. T., & Sarama, J. (2001). Logo and geometry. *Journal for Research in Mathematics Education, Monograph*, 10, i-177.
- Cohen, J. W. (1988). *Statistical power analysis for the behavioural sciences*. Hillsdale: Lawrence Erlbaum Associates.
- Confrey, J., & Lachance, A. (2000). Transformative teaching experiments through conjecture-driven research design. In A. Kelly & R. A. Lesh (Eds.), *Handbook of research design in mathematics and science education* (pp. 231–265). Mahwah, NJ: Lawrence Erlbaum Associates.
- Cooper, T., & Warren, E. (2008). The effect of different representations on years 3 to 5 students' ability to generalise. *ZDM Mathematics Education*, 40(1), 23–37.
- Cooper, T. J., & Warren, E. (2011). Years 2 to 6 students' ability to generalise. In J. Cai & E. Knuth (Eds.), *Early algebraization* (pp. 187–214). Berlin: Springer.
- Creswell, J. (2008). *Research design: Qualitative, quantitative, and mixed methods approaches* (3rd ed.). Thousand Oaks, CA: SAGE Publications.
- English, L. D. (2016). STEM education K-12: Perspectives on integration. *International Journal of STEM Education*, 3(1), 1–8.
- García-Peñalvo, F. J., Reimann, D., Tuul, M., Rees, A., & Jormanainen, I. (2016). *An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers*. Belgium: TACCLE 3 Consortium.
- Han, S., & Bhattacharya, K. (2001). Constructionism, learning by design, and project based learning. In M. Orey (ed) *Emerging perspectives on learning, teaching, and technology*. <http://pirun.ku.ac.th/~btun/papert/design.pdf>. Accessed 10 Sep 2019.
- Hattie, J. A. C. (2009). *Visible learning: A synthesis of over 800 meta-analyses relating to achievement*. London: Routledge.
- Highfield, K. (2015). Stepping into STEM with young children: Simple robotics and programming as catalysts for early learning. In C. Donahue (Ed.), *Technology and digital media in the early years* (pp. 150–161). New York: Taylor & Francis.
- Hoyles, C. (1985). Developing a context for LOGO in school mathematics. *The Journal of Mathematical Behavior*, 4(3), 237–256.
- Hoyles, C., & Noss, R. (1992). A pedagogy for mathematical microworlds. *Educational studies in Mathematics*, 23(1), 31–57.
- Lesh, R., & Lehrer, R. (2000). Iterative refinement cycles for videotape analyses of conceptual change. In R. Lesh & A. Kelly (Eds.), *Research design in mathematics and science education* (pp. 665–708). Hillsdale, NJ: Erlbaum.
- Lewis, C. M., & Shah, N. (2012). Building upon and enriching grade four mathematics standards with programming curriculum. In *Proceedings of the 43rd ACM technical symposium on computer science education* (pp. 57–62). ACM.
- Lüken, M. M. (2018). Repeating pattern competencies in three-to five-year old kindergartners: A closer look at strategies. In I. Elia, J. Mulligan, A. Anderson, A. Baccaglini-Frank, & C. Benz (Eds.), *Contemporary research and perspectives on early childhood mathematics education* (pp. 35–53). Cham: Springer.
- Meng, C., Idris, N., Leong, K. E., & Daud, M. (2013). Secondary school assessment practices in science, technology and mathematics (STEM) related subjects. *Journal of Mathematics Education*, 6(2), 58–69.
- Miller, J., & Larkin, K. (2017). Using coding to promote mathematical thinking with year 2 students: Alignment with the Australian curriculum. In A. Downton, S. Livy, & J. Hall (Eds.), 40 years on: We are still learning! (Proceedings of the 40th annual conference of the mathematics education research group of Australasia, (pp. 469–476). Melbourne: MERGA.
- Miller, J., & Warren, E. (2012). An exploration into growing patterns with young Australian Indigenous students. In J. Dindyal, L. P. Cheng, & S. F. Ng (Eds.), *Mathematics education: Expanding horizons* (Proceedings of the 35th annual conference of the Mathematics Education Research Group of Australasia) (pp. 505–512). Singapore: MERGA.
- Moore, T. J., Stohlmann, M. S., Wang, H., Tank, K. M., Glancy, A. W., & Roehrig, G. H. (2014). Implementation and integration of engineering in K-12 STEM education. In S. Purzer, J. Strobel, & M. Cardella (Eds.), *Engineering in pre-college settings: Research into practice* (pp. 35–60). West Lafayette, IN: Purdue University Press.
- Mulligan, J., & Mitchelmore, M. (2009). Awareness of pattern and structure in early mathematical development. *Mathematics Education Research Journal*, 21(2), 33–49.

- Noss, R. (1986). Constructing a conceptual framework for elementary algebra through Logo programming. *Educational Studies in Mathematics*, 17(4), 335–357.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books Inc.
- Papert, S., & Harel, I. (1991). *Constructionism*. New York: Ablex Publishing Corporation.
- Papic, M. (2007). *Mathematical patterning in early childhood: An intervention study*. Unpublished PhD thesis, Macquarie University.
- Papic, M. M., Mulligan, J. T., & Mitchelmore, M. C. (2011). Assessing the development of preschoolers' mathematical patterning. *Journal for Research in Mathematics Education*, 42(3), 237–268.
- Partnership for 21st Century Skills (P21) (2019). *Framework for 21st century learning*. <http://www.battelleforkids.org/networks/p21>. Retrieved June 30, 2019.
- Premsky, M. (2008). *Programming is the new literacy*. <http://www.edutopia.org/literacy-computer-programming>. Retrieved February 12, 2018.
- Prinsley, R., & Johnston, E. (2015). *Transforming STEM teaching in Australian primary schools: everybody's business*. Australian Government: Office of the Chief Scientist.
- Rittle-Johnson, B., Fyfe, E. R., McLean, L. E., & McEldoon, K. L. (2013). Emerging understanding of patterning in 4-year-olds. *Journal of Cognition and Development*, 14(3), 376–396.
- Savard, A., & Highfield, K. (2015). Teachers' talk about robotics: Where is the mathematics? In M. Marshman, V. Geiger, & A. Bennison (Eds.), *Proceedings of the 38th Annual Conference of the Mathematics Education Research Group of Australasia* (pp. 540–546). Sunshine Coast: MERGA.
- Stahl, B.C. (2003). *How we invent what we measure: A constructionist critique on empiricist bias in research*. Paper presented at the 9th Americas conference on Information Systems, Florida.
- Steffe, L. P., & Thompson, P. W. (2000). Teaching experiment methodology: Underlying principles and essential elements. In R. Lesh & A. E. Kelly (Eds.), *Handbook of research design in mathematics and science education* (pp. 267–306). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Sullivan, P. (2011). *Teaching mathematics: Using research-informed strategies*. Camberwell: ACER.
- Threlfall, J. (1999). Repeating patterns in the early primary years. In A. Orton (Ed.), *Pattern in the teaching and learning of mathematics* (pp. 18–30). London: Continuum.
- Vasquez, J., Sneider, C., & Comer, M. (2013). *STEM lesson essentials, grades 3–8: integrating science, technology, engineering, and mathematics*. Portsmouth, NH: Heinemann.
- Voogt, J., & Roblin, N. P. (2012). A comparative analysis of international frameworks for 21st century competences: Implications for national curriculum policies. *Journal of curriculum studies*, 44(3), 299–321.
- Warren, E., & Cooper, T. (2007). Repeating patterns and multiplicative thinking: Analysis of classroom interactions with 9-year-old students that support the transition from the known to the novel. *The Journal of Classroom Interaction*, 2(1), 7–17.
- Warren, E., & Miller, J. (2013). Young Australian Indigenous students' effective engagement in mathematics: The role of language, patterns, and structure. *Mathematics Education Research Journal*, 25(1), 151–171.
- Warren, E., Miller, J., & Cooper, T. (2012). Repeating patterns: Strategies to assist young students to generalise the mathematical structure. *Australasian Journal of Early Childhood*, 37(3), 111–120.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Zazkis, R., & Liljedahl, P. (2002). Generalization of patterns: the tension between algebraic thinking and algebraic notation. *Educational Studies in Mathematics*, 49, 379–402.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.