Art. #1959, 9 pages, http://www.sajournalofeducation.co.za

Foundation Phase teachers' experiences of teaching the subject, coding, in selected Western Cape schools

Cindy Jean Geldenhuys
Rondebosch Boys' Preparatory School, Rondebosch, South Africa
Aslam Fataar
Department of Education Policy Studies, Faculty of Education, Stellenbosch University, Stellenbosch, South Africa
afataar@sun.ac.za

Several teachers have recently started introducing coding into their teaching in primary schools. This comes on the back of the emerging prominence of educational technology and the teaching of computational skills at school level, in light of the country's policy commitment to the Fourth Industrial Revolution. Coding has been punted as one of two essential subjects to be launched in schools countrywide from 2020; the other being robotics. In this article we focus on the implementation of coding as a subject in selected Foundation Phase classes in the Western Cape. We aim to gain an understanding of coding as a subject from the perspective of teachers who are implementing this very new subject in the Foundation Phase. We specifically discuss the experiences and challenges of teachers who have been teaching the subject over the last few years, based on in-depth qualitative interviews with four Foundation Phase teachers. Overall, we provide a set of considerations for the optimal implementation of coding as a subject in Foundation Phase in South African schools. The participants' experiences highlight the challenges associated with implementation, teachers' pedagogical skills and competences, and resource requirements. We raise the following areas that need to be addressed for the successful implementation of coding: professional development addressing teaching methodologies on the development of computational thinking skills in young learners, providing support for teachers, addressing time constraints in the teaching of the subject, and providing resources.

Keywords: coding; coding subject teaching; Foundation Phase; teachers; Western Cape schools

Introduction

Globally, coding has increasingly gained attention in education spheres over the last few years. What has long been a powerful language for software developers, coding has since come to be considered by some as crucial for "21st century literacy," on par with reading, writing and numeracy. It is believed that without coding skills students would be left behind when they attempt to enter the workforce. This has prompted a wave of curriculum changes and coding is now taught from an early age in many countries. In light of the country's policy commitment to the Fourth Industrial Revolution and in an attempt to ensure its learners do not get left behind, South Africa's Department of Basic Education (DBE) announced that coding and robotics would be introduced in Grades R to 9 in all schools. The implementation of coding will begin with a pilot project in 2020 in 1,000 schools across five provinces in Grades 7 to 9 (Motshekga, 2019). The introduction of coding in schools has, however, been the subject of contentious debate. Reports in the media reveal that there is scepticism about whether the introduction of coding in South African schools would be successful, given the challenging teaching conditions and weak basic reading, writing, and mathematics skills among learners. Given these challenges, questions have been raised about whether focus should be diverted from addressing these issues.

Despite this scepticism, a small number of Foundation Phase teachers have started teaching Coding in their schools. This has prompted us to embark on research that explored Foundation Phase teachers' experiences of teaching coding within their classroom context. In this article we present a discussion of the views of four selected teachers in Western Cape schools who were teaching coding in the Foundation Phase at the time of the study. The article is based on our research, which responded to the following question: What are Foundation Phase teachers' experiences, practices and perspectives of teaching the subject, coding? The article provides an evaluation of the practices and challenges with respect to the implementation of the subject, coding, in Foundation Phase classrooms in the Western Cape, and offers suggestions for improving schools' and teachers' capacities for teaching the subject.

Literature Review

Over the last two decades, traditional educational practices have been challenged to prepare learners for an increasingly digital society, which ushered in an era of technology use in education. Discussion on the effectiveness of educational technology in improving teaching and learning is prominent in the literature and, despite a lack of consensus, technology-related educational changes are ubiquitous across the world (Mao, Ifenthaler, Fujimoto, Garavaglia & Rossi, 2019:284–291). The rise of educational technology has not only encouraged new pedagogical approaches; it has also prompted curriculum changes and the introduction of new subjects such as coding and robotics.

Following global trends, the subject being introduced in the General Education sector in South Africa is called coding. Ching, Hsu and Baldwin (2018:564), however, make a useful distinction between coding and

computer programming. They suggest that programming can be defined as the provision of instructions to execute a procedure or task to solve a problem, while coding entails writing a set of instructions in a specific programming language that the machine understands (Ching et al., 2018:565). Malik and Coldwell-Neilson (2017:1090) explain that novices have to learn both the syntax and semantics of a programming language while developing problem-solving skills. Zaharija, Mladenović and Boljat (2013:1577) propose that learners be exposed to basic programming concepts from an early age. Much like teaching reading, where teachers begin by developing learners' phonemic awareness and teaching letter recognition, when teaching programming, there are foundational skills that should be developed prior to teaching programming languages. Most literature on the teaching of coding and programming to young learners supports the development of foundational skills of computational thinking, rather than focusing solely on teaching learners to write lines of code in a specific programming language (see Buitrago Flórez, Casallas, Hernández, Reyes, Restrepo & Danies, 2017; Chalmers, 2018; Shute, Sun & Asbell-Clarke, 2017; Wing, 2006; Yadav, Krist, Good & Nadire Caeli, 2018).

A consensus on the definition of computational thinking in the educational context, however, remains unresolved and continues to evolve (Denning, 2017; Yadav et al., 2018:374). The current wisdom holds that computational thinking encompasses a set of higher-order thinking skills derived from computational logic, which can be applied to solve problems (Buitrago Flórez et al., 2017:834; Chalmers, 2018:93–94; Hsu, Chang & Hung, 2018; Papert, 1980; Shute et al., 2017; Wing, 2006; Yadav et al., 2018:371–374).

Angeli, Voogt, Fluck, Webb, Cox, Malyn-Smith and Zagami (2016) developed a framework for introducing children (aged 6–12) to computational thinking concepts. The framework identifies five skills to be developed across different subject areas, namely, abstraction, generalisation, decomposition, algorithmic thinking, and debugging (Chalmers, 2018:94; Shute et al., 2017). According to the framework

> students develop the skills of abstraction and generalization from one solution to another by identifying familiar patterns; develop decomposition skills as they break down complex problems into solvable chunks; and use algorithmic thinking to devise sequences of actions to be executed. The iterative problem-solving process also involves students using debugging skills as they identify and fix issues and errors (Chalmers, 2018:94).

Nardelli (2019) argues that computational thinking skills are indistinguishable from skills that are already included in school curriculum subjects and have been developed in young learners in schools for many years. Nardelli (2019) thus advocates that the development of computational thinking skills should be more explicitly integrated into other subjects. Whether integrated or not, however, the question of how to teach computational thinking in the most effective manner remains. A growing number of educators have started introducing computational thinking skills (Ching et al., 2018:563), the majority of whom use coding as the primary means to develop computational thinking among their learners (Yadav et al., 2018:371). In most cases coding is taught by computer teachers as a separate subject during traditional information and communications technology (ICT) lessons.

A review of the literature shows that through programming, learners learn how to solve problems in systematic ways and develop and exercise higher-order thinking skills such as problem decomposition, analysis, and evaluation, which are critical to problem-solving (Ching et al., 2018:564). Learners who are exposed to computational thinking through programming, develop algorithmic thinking, problem-solving, logic, and debugging skills (Buitrago Flórez et al., 2017:837). Based on findings, the view that teaching programming is the best approach to teach computational thinking has become accepted. This is only the case, however, if curricula are centred around the development of computational thinking skills and not solely focused on teaching children coding languages (Buitrago Flórez et al., 2017:837). The ability to write lines of code is regarded simply as the vehicle through which the learners can create stories, animations, objects, mobile applications (apps) or games and solve problems. It is during teachers' planning, execution and improvement of these activities that computational thinking is developed in young learners.

The introduction and use of programming as a vehicle to develop computational thinking in primary schools, however, brings into question the developmental readiness of young learners for an activity that involves highly abstract concepts and learning many skills at once. It raises the question of how one teaches programming when learners have not developed the ability to think abstractly and logically as is required when programming. According to Piaget's theory of cognitive development, primary school children are entering the concrete operational stage of cognitive development, which is characterised by the development of logical thinking and problem-solving skills (Zaharija et al., 2013:1577). To effectively use programming in primary schools where children are entering the concrete operational stage, Zaharija et al. (2013:1577) propose that,

> … it is necessary to develop an approach to teaching that would make programming more accessible to primary school learners and would encourage the development of their logical thinking and problem-solving, but at the same time would still be appropriate for their age.

Strawhacker and Bers (2019) identified the following two areas of programming knowledge that they believe teachers should concentrate on for learners in the five to seven age group. One is the ability to match a programming command with its outcome or action. The second is the ability to construct a program that uses the correct commands in the correct order. They indicate that if taught in developmentally appropriate ways, primary school learners are not too young to learn programming. Increased interest in introducing young learners to programming has led to innovative developments in educational technology that have made programming more accessible and less abstract for young learners. Many simplified child-friendly programming languages have been developed. Several new visual programming languages make use of graphical elements such as blocks and puzzle pieces that users can manipulate and drag-and-drop rather than having to write text-based code. These include the popular programming environments Scratch and Tynker. These graphical programming environments facilitate learning of programming language syntax and semantics needed in order to execute commands. These environments usually include simulation and provide quick feedback on work, allowing learners to rapidly learn from their actions and iterate on their designs. The graphical languages also make abstract concepts more easily accessible by using pre-programmed building blocks that simplify the logic required to implement these functions. The drive to make coding more accessible to younger learners has also led to the development of board games, electronic blocks, devices controlled by buttons, children's storybooks, and sticker books (Ching et al., 2018:564).

As computational thinking in education gains popularity, many governments across the globe have started expecting teachers to teach coding from a young age (Rich, Browning, Perkins, Shoop, Yoshikawa & Belikov, 2019:311–312). A study by Rich et al. (2019) aimed to better understand the profiles, practices and problems of those who are teaching coding to younger children. They surveyed 313 teachers in the United States of America (USA), United Kingdom (UK), Finland and Australia whose previous experience of teaching programming ranged from as low as one year to as much as fifteen years. More than 55% of the teachers had little or no training with programming prior to deciding to teach it in the classroom. Just over half the teachers stated that they chose to teach programming, while the other half stated that they were required to do so at their schools. The Scratch programming environment proved to be the most commonly used tool. The UK was the only country in the study where a text-based language (Python) was commonly taught. Teachers reported that they taught programming as a standalone subject nearly twice as often as an integrated subject. When teachers taught coding as an integrated topic, they found that coding was most often integrated with mathematics, followed by science, language, engineering, and social science (Rich et al., 2019:319).

The teachers in the study by Rich et al. (2019) were overwhelmingly encouraging towards other teachers starting to teach programming, regardless of whether they had programming knowledge or not. A number of the teachers said that teachers should be willing to learn with and from their students when they started out teaching it. When asked what they would do differently their responses were largely in line with the literature and included statements like “learn the theory first, instead of starting with straight coding”, “introduce more computational thinking concepts rather than just code”, and “teach the concepts before the code” (Rich et al., 2019:327). Similarly, the research on which this article is based attempted to gain insight into the experiences of South African Foundation Phase teachers who were teaching coding.

Methodology

We adopted a qualitative research design that allowed us to investigate the experiences of Foundation Phase teachers who were teaching coding in their classroom contexts. Based on observing ethical protocols, including the guarantee of anonymity and the right of withdrawal, the research took place at selected schools in the Western Cape. An effort was made to select teachers from a range of school contexts, including schools in all quintiles, and we searched for teachers across the socioeconomic class spectrum. Despite efforts, we were not able to find Foundation Phase teachers from working class schools who have introduced coding into their teaching.

The participants who were interviewed were four teachers from four different schools identified via purposive and snowball sampling. All the schools followed the CAPs curriculum, albeit one of the schools was an independent school. The participants had varying levels of teaching experience, ranging from five to thirty years. All participants were female and three of the four had no previous exposure to computer programming prior to teaching computational thinking and coding.

We interviewed each of the participants once, using semi-structured interviews. Each interview

lasted between 45 and 60 minutes and was recorded and transcribed. We used the constant comparative method of data analysis, concentrating on an analysis of the content of the interviews. Once the transcriptions were verified by the participants, we sorted and coded the data into different categories and themes with similar units of meaning. While coding, we made notes of questions and thoughts that arose. Once all the data had been coded, we integrated the categories of analysis and began looking for patterns that emerged, again making notes throughout this process. We simultaneously coded and analysed the data, which allowed for a more fluid development of themes from the data than would have been possible if the coding and analysis were each completed in isolation. From the data analysis we hoped to gain an in-depth understanding of selected Foundation Phase teachers' experiences, practices and perspectives of teaching coding.

Data Presentation and Analysis

The analysis of the participants' interviews presented below aims to provide insight into Foundation Phase teachers' experiences teaching coding. The findings are organised into four main themes: the teachers' background information and school contexts; teachers' skills, training and support; teachers' views of learners' skills; and their teaching practices and experiences.

We were aware that the key limitation of our qualitative research approach was the small sample size. We were aware that the findings could not be generalised to coding teachers in primary school contexts. This research avoids a one-size-fits-all practical significance given the diversity of the schools in which the teaching of the subject takes place. The results must thus be interpreted with caution. We offers insight into the four selected coding teachers' teaching practices which should enrich debate about the subject's implementation in schools.

Teachers' Background and School Contexts

The four participants held formal teacher education qualifications. Three of them held Bachelor of Education degrees specialising in Foundation Phase, while the fourth held a Higher Diploma in Education. Two of the participants have five years of teaching experience, one had fourteen and the fourth, thirty years. They started introducing coding into their teaching within the last three years. None of the participants had previous coding knowledge or computer programming experience prior to teaching it and no training was provided during their formal studies. Despite the lack of prior knowledge of coding, the participants were fairly confident when they began introducing it to their teaching. Only one participant remarked that "I felt very confused by it, because I had never done any coding so I just fumbled my way [through]". This general confidence may be because the participants were mostly making use of tools such as websites or board games to teach coding. According to the four teachers, these tools have proven to be fairly simple to use.

Interviewees were asked whether teaching coding was required of them by their school or whether it was optional. All of them responded that it was optional and that they were self-driven to teach the subject. Once they started and showed some success, they were encouraged by their school management to continue teaching it. The journey that led to introducing coding into their teaching was similar for all four participants. They were all actively using educational technology in their teaching. They attended educational technology workshops, courses and conferences as part of their professional development where they were introduced to the idea of teaching coding to young learners. One participant said the following of the conferences: "*I went to conferences and people were just talking as if it is happening in schools and I started sweating and panicking and* [thinking] *but we need to do this and I went back to school and I told my principal, look here, everybody out there is doing this*". The workshops, courses and conferences were pertinent in sparking their interest in coding, and many admitted that because of the conferences they became convinced of the importance of young children learning coding.

While reflecting on her decision to start teaching coding initially, one participant explained that she "*started introducing coding as a fun activity because I'm just one of those teachers. I've been in the profession forever, but I just love anything new.*" She also mentioned that her school has a robotics club for Intermediate Phase learners. The robotics teams were very successful and have won competitions. As Foundation Phase teacher, she wanted to expose learners to coding and robotics from a young age to prepare them for when they were old enough to be a part of the robotics team.

Teacher Skills, Training, and Support

Responses to questions about the skills that teachers needed to teach coding suggested that all the participants felt that few new skills were required, but that obtaining some new knowledge was important. For one participant this knowledge included some understanding of coding terminology which is used in the board game, Scottie Go. To learn this, she watches videos on YouTube and used free online resources. Another participant felt that it was necessary for teachers to,

> *... know what coding is because, I think, there are a lot of teachers that don't understand what coding means, they think it's just a*

> *bunch of numbers.* [...] *I think people need to understand 'this is how it looks, this is what you expect from a grade one level' and then to work from that with the skills. I think normal, you know, computer skills is necessary and an understanding of the program and what they are working from.* [*sic*]

This participant stated that her objectives for the learners were for them to have an understanding of things "*like the syntax and run, and what a function is and* [...] *what a command is*", which implies that she thinks it is necessary for teachers to have this understanding first. It is possible that the participants' view that few skills were required, stemmed from them using simplified applications such as online board games or simplified robotic devices, which are fairly user-friendly.

Only one of the participants referred to the need for teachers to know how to develop computational thinking skills in their teaching. This person mentioned that was is necessary for teachers to have a sound understanding of coding concepts, such as functions, loops and conditionals. She said that although one would not necessarily initially explicitly teach these concepts to Foundation Phase learners, it was necessary to have an understanding of them as these were key to developing learners' computational thinking. She further remarked that if teachers followed a formal computer programming curriculum in their class teaching, based on dedicated and properly scaffolded lessons each week, it was possible to introduce these concepts formally. She commented as follows when she explained what she did with her Grade 3s:

> *My Grade 3s are able to explain what a condition is and when you would use it. They have used 'if' and 'if-else' statements to create games. For example, they programmed a character to follow the player's mouse to be moved from the start to the finish of a maze they designed. To complete the game the player would have to move through the maze without touching the sides. If the sides were touched the game was programmed using an 'if' statement to say 'Game over' and end. If the player reached the finish, the learners programmed their games to say 'Well done.' While working on games on Scratch, I introduce and explain these different coding concepts to learners, so it is important that I understand them myself.*

She felt that it was important for teachers to know how to debug, as well as guide learners in picking up bugs themselves when programs do not work.

With regard to skills development, all the participants held courses, conferences and workshops in very high regard. One participant had this to say about the courses: "*We always make sure that we are there whenever things happen*". In order to acquire the knowledge required to teach coding, all the participants took online courses and attended conferences, courses and workshops, usually paid for by their schools.

Three participants received no training or support from the Western Cape Education Department (WCED). The fourth participant mentioned that she had attended two conferences organised by the WCED that focused on integrating educational technology into teaching, where coding was one of the aspects covered. Three of the participants had received some training and guidance from independent educational technology consulting companies at the start of their journey.

Two of the participants felt that their school management teams (SMTs) did not provide adequate support in teaching coding even though they encouraged them to continue teaching the subject. This may imply that the SMTs did not have an adequate understanding of the training requirements of the teachers. However, the other two participants felt that their schools consistently supported the efforts of teachers to start introducing coding into their teaching.

When asked who they turned to for support when challenges arose. One participant said that she often went to the school's information technology (IT) technician for help, and another explained that she turned to her software developer husband if she needed support. A third participant preferred doing things alone, and the fourth participant expressed her frustration about how difficult it was to find fellow teachers with whom to discuss challenges. She mostly had coding related conversations with Intermediate Senior Phase teachers at other schools whom she met at conferences and workshops. A participant spoke about how she and another teacher from a neighbouring school often share resources with each other. The participants expressed the need for a professional learning community of teachers where information, ideas and perspectives could be shared for more effective teaching of the subject.

Teachers' Views on Learners' Readiness to Learn Coding

In this theme we discusses the teachers' views about the learners' readiness and capacity to learn coding, which emerged as crucial to the way that the teachers approached their teaching of the subject. The teachers were all positive about their learners' cognitive readiness for the type of coding activities they were being introduced to at that early age. In contrast, when asked what skills they thought were important for learners to acquire as a precursor to teaching coding, their answers were vague. Instead, participants were able to answer what skills they hoped to develop while learners engaged in coding-related activities. Surprisingly, problem-solving skills were not raised as one of the primary skills. Participants did note, however, that there was a change in the way that the learners approached problems and that learners showed an increased determination to solve problems despite

finding it frustrating or challenging. Participants also found it interesting how learners found resourceful ways to solve problems, such as asking older Grade 7 learners in the robotics team for help. The teachers felt strongly that they were developing collaborative learning, teamwork and communication skills.

Three of the participants linked their objectives for teaching coding to their mathematics teaching, specifically in relation to spatial skills such as position, direction and other map skills. One participant explained that she felt that the skills that should be emphasised for teaching coding in the Foundation Phase were in effect skills which were already outlined in the curriculum. She believed that coding could be taught as part of other subjects "*without them* [the learners] *knowing how coding fits in*".

One participant aimed to prepare learners for the text-based programming environment that they would work with in their programme from Grade 4 onwards. She said that she wanted them to have an understanding of "*the syntax and run, and what a function is and* [...] *what a command is*", and to understand that coding is a language, and that there are various different coding languages. The focus for this teacher was on preparing learners to write lines of code rather than developing the computational thinking skills needed for computer programming.

Another participant alluded to computational thinking skills when she mentioned that her aim was to develop learners' sequential thinking skills and the ability to provide step-by-step instructions to solve problems. This participant explicitly developed learners' understanding of coding concepts, such as functions, loops and variables, through the use of game development. She also aimed to develop learners' debugging skills and encouraged them to identify the problems themselves.

Another participant mentioned that she thought that games not directly related to coding, like chess, were important. Likewise, another participant said that she had tried to encourage the Grade R teachers at her school to start playing more with Lego and encouraged all Foundation Phase teachers to build things more often. These responses showed that the participants were starting to understand that the teaching of coding required the development of computational and higher-order thinking skills, but that they were yet to reflect on this and apply it to their practice consciously. The conferences and courses which the participants attended for professional development and training, also seemed to have failed to show how computational thinking can be embedded in authentic learning situations in other content areas.

Practices and Experiences of Teaching Coding

In this section we discuss the practices and experiences of the four interviewees with respect to teaching coding. For the first few lessons, the participants taught coding as an isolated subject. Three of the participants decided early on to integrate it into their teaching of other subjects, namely mathematics, English Home Language and life skills, although they also taught it as a stand-alone subject. The other participant encouraged fellow teachers at her school to integrate coding into their teaching of other subjects but found it difficult to do so in her computer lessons.

Participant 1

Participant 1 used drag-and-drop games, such as the Hour of Code games on the Code.org website, in her classes. Given her focus on computer literacy skills, however, this did not happen very often. In her role as the educational technology coach, she visited Grade R and 1 classes once or twice a term to do demonstration lessons for the teachers and to show them how to use Pro-bots. A Pro-bot is a rechargeable floor robot which can be programmed sequentially using the numerical keypad and arrows or using the computer software. The Pro-Bot has a built-in pen mechanism that can be used to draw. This participant demonstrated to her colleagues how Pro-bots could be used in lessons on shapes. When she wanted the teachers to start using tablets more often in class, she demonstrated lessons using Spritebox. Spritebox is a children's coding app where the Sprite needs to be given a sequence of commands to make it through a maze to free up the Sprite's bottled-up friends.

Participant 2

Participant 2 mostly did coding activities while doing small-group work with her Grade 2s. While she was busy with a group on the mat teaching them a specific maths concept, and the rest of the class was at their tables doing a worksheet, one group of learners would be playing the coding board game, Scottie Go. Scottie Go is described on their website as "a combination of cardboard tiles, which are used by the players to create programming commands, and an app that sets tasks and scans the proposed solutions that set Scottie and other characters in motion".

Towards the end of the term, this participant usually linked up the sequencing that the learners did while playing Scottie Go, with her mathematics or life skills content. She commented on how well it linked to "direction, left, right, mapping", but she also admitted that she did not have the time to focus on coding explicitly. For the most part, she expected the learners to play the board game independently. She used more competent learners in the class to monitor and assist those who were

struggling so that she was not disturbed while teaching on the mat. She also said that she occasionally used coding activities "as a brain session" for learners who needed to refocus before going back to their classwork. She also let her learners play chess during class.

Participant 3

The third participant taught coding classes to Grade 3s and described that a typical lesson would consist of her showing the learners a game that she has created. As a class, they discussed the game and broke it up into smaller sections. Learners would then be given specific challenges or tasks to assist them in slowly creating their own games, similar to the one that was shown to them, section by section. These games were made on Scratch and usually took about eight weeks to complete. Throughout the tasks they focussed on and explained different coding concepts. Examples of games included balloon popping games, mazes and Snakes and Ladders. Learners were often given the freedom to create their own animations and interactive stories on ScratchJr by programming characters to perform certain actions. In Participant 3's previous position as a class teacher, a typical coding lesson was integrated into small-group teaching on the mat for reading or mathematics and involved the use of the Code and Go Robotic Mice. Learners would have to programme the mouse's path using arrow buttons to navigate different obstacles and to get to the correct sight words or answers to sums or word problems. Participant 3 also tried to encourage the development of mathematical thinking skills by posing carefully thought-out word problems to the learners during small-group teaching on the mat.

Participant 4

Participant four, who taught Grade 3, followed a similar approach to Participant 2, using Scottie Go and teaching in small groups while another group was playing Scottie Go independently. She remarked that she did not have the time to teach the learners how to play Scottie Go. Instead, she taught one group of learners and then split them up and expected of them to teach the rest of the class. Participant 4 also asked learners to create their own sequential games. She described her intentions as follows:

> *Make sure every class has a robot, I would think, and then* [ask] *what is this? What is happening? How does this robot know to move left, right, forward? Then, if possible, just take it apart and then build it up from scratch. You know it's stuff like that, because that is really what will get them interested in it.*

Participant 4 also discussed how she developed learners' skills, which were not necessarily coding skills, but to prepare learners for when they started with it as a formal subject in later grades. She described in particular how playing position games on the interactive whiteboard could be linked to the skills needed for coding. For example, she explained how directing a pirate to find the treasure or a bug to food on a grid map could develop the precursor skills needed for coding.

In addition, Participant 4 developed a word wall which she called the Future Wall. This was to develop learners' vocabulary for words related to technology and coding. She added words from the board game based on coding concepts as well as names of apps that they used as a class. She regularly discussed with her learners "*that this is the language you are going to be speaking in the future*."

Besides the board games and robotic devices mentioned, all the participants mentioned the use of tablets, Chromebooks, cellular phones or desktop computers. One participant had to provide her own iPad and cellular phone for the learners to use, but the other schools provided devices for the participants. In most cases, apps and games, where the learners had to provide directional instructions for a character to navigate a maze, were played on these devices. Games mentioned included LightBot, Alice, Spritebox and the Hour of Code games on Code.org. With her Grade 3s, Participant 3 uses ScratchJr and Scratch.

In preparation for coding lessons, other resources were used as well. One participant followed a formal curriculum developed and provided by an independent company. The curriculum includes lesson plans, training videos and on-site training when needed. Before moving to a new school, however, this participant did not follow a formal curriculum, much like the rest of the participants. Two other participants referred to free online curricula that they followed informally. The curriculum on the website, http://www.code.org/, is an example of one of these online platforms.

With the interviews we tried to gain insight into teachers' formal teaching methodologies and assessment practices employed when teaching coding. What we found was that very little focus was on teaching methodologies. Some of the comments made when asked about teaching methodologies included: "*I suppose I haven't dug too deeply in methodologies*", "*at the moment that is all that I am capable of doing with school*", and "*I haven't consciously thought about it but I suppose I use the same teaching methodologies as I would when teaching any other Foundation Phase subject*". The participants thus did not seem to think systematically about their coding teaching methodologies. From the discussions though, it could be deduced, that problem-based learning, gamification, collaborative learning, flipped classroom teaching and peer teaching were examples of teaching methodologies employed by the participants when teaching coding.

From the interviews, one could conclude that since coding did not form part of the formal curriculum, participants did not feel that it was necessary to monitor learners' progress and development formally. Three participants spoke about how they informally observed the learners while they were doing activities. One of the participants started assessing learners' skills once she started teaching coding more formally. She used ScratchJr's Solve It tasks. These assessments provided different ways to determine the depth of the learners' understanding of the relationship between the programming blocks and their associated behaviours. The Solve It tasks have been used in research to assess children's programming knowledge and understand children's logic while answering questions (Strawhacker & Bers, 2019).

The participants felt that teaching coding in the Foundation Phase in the South African context had its own challenges. The biggest challenge raised by everyone was a lack of time. They all expressed that they struggled to make time to teach coding, given the fact that it already was a challenge to teach everything that was expected of them in other subject areas. One participant shared how it was particularly difficult to find time to integrate coding into her teaching since the majority of the learners in her class were not yet reading at the acceptable grade level. She often felt tension over whether focusing on coding was reducing the time available to focus on basic literacy skills. The lack of resources was also raised as a challenge. In many cases, the teachers had to find ways to use the resources effectively so that all learners could get a chance to use them, and two of the participants expressed the desire to have more resources. The noise levels, while learners were working together in groups and trying to solve the challenges or play board game, was raised as a concern by two participants. One participant mentioned that, at times, the free software or programmes being used were frustrating. Often what would appear to be free would only be for a few levels, and once learners had played a few rounds, the free trial would end. Interestingly, none of the participants mentioned WiFi and connectivity as a challenge.

Despite these challenges, the participants' general experiences of teaching coding had been positive. They expressed how well it was received by the learners and how much they enjoyed it. Participant 4 said: "*it's just that the children nag us so much* [to do it] *that I think we don't have a choice*." According to them, their SMTs have also responded to the introduction of coding well despite the teachers not being expected to teach it. Their colleagues were positive, but showed little intention of starting to teach coding themselves.

Conclusion

With this research we aimed to gain insights into selected Foundation Phase teachers' experiences, practices, and perspectives of teaching coding in schools in the Western Cape. We set out to better understand teachers' experiences leading up to, and their teaching of coding, the resources they used, the skills they believed teachers required to teach coding, and the skills they had hoped to develop in learners. Four teachers with varying levels of teaching experience were interviewed using semi-structured interviews. The participants' level of teaching experience seemed to have little influence on their experiences of teaching coding. All participants started teaching coding in the last three years and had no prior coding knowledge. The participants initiated and introduced coding on their own since they were convinced of its benefits and were concerned that their learners would be left behind. Generally, they used games to teach coding, as a stand-alone subject and integrated into other subjects.

Even though participants' intentions were good and they were confident in their teaching of coding, their interview responses in comparison to the literature revealed that they lacked the pedagogical skills to teach the foundational skills required for coding. The biggest hindrance to teaching coding effectively proved to be the lack of awareness of computational thinking skills and a reflection on the precursor skills that learners would need to be able to participate effectively in coding.

No participants received school-based or departmental training but pursued their own professional development. Despite the courses and training attended, interviewees' responses revealed that they had spent very little time reflecting on teaching methodologies and assessing learners' development. Most of the participants provided only vague ideas of the skills required to teach coding and could not give concrete answers about learner skill requirements or developmental progression.

Time constraints also proved to be a hindrance. In addition, coding lessons proved to be resource-intensive and required some participants to personally purchase resources such as online games, board games and robotic devices, with most schools only providing computers.

The findings from the research suggest that in their pursuit of the implementation of coding as a subject in the Foundation Phase, the DBE needs to provide training for teachers and School Management Teams that focuses on teaching methodologies on the development of computational thinking skills in young learners. For the subject to be optimally implemented, support

structures would need to be set up, time constraints would need to be addressed and resources would need to be provided.

With this article we have provided an important perspective on the implementation dynamics and challenges faced by teachers involved in the teaching of coding in the Foundation Phase. Further studies are needed to gain a clearer understanding of teachers' experiences, practices and perspectives. Such studies should concentrate on teachers who teach coding in more diverse school contexts as this would assist educational planners and departments to develop a coding curriculum for primary schools in South Africa.

Authors' Contribution

Cindy Jean Geldenhuys did the research on which this manuscript is based. It was co-authored by Cindy Jean Geldenhuys and Aslam Fataar. Both authors reviewed the final manuscript.

Notes

i. Published under a Creative Commons Attribution Licence.
ii. DATES: Received: 4 December 2019; Revised: 28 August 2020; Accepted: 5 October 2020; Published: 30 November 2021.

References

Angeli C, Voogt J, Fluck A, Webb M, Cox M, Malyn-Smith J & Zagami J 2016. A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, 19(3):47–57. Available at https://www.jstor.org/stable/pdf/jeductechsoci.19.3.47.pdf?casa_token=8VKhy8D6U2gAAAAA:FlpXMtjdZmxtf_BeetS1xYDciepw2e7Nsb9oWobmhhap06uqpWPtNlVNhOWnfoYo6gXwXvrMTfhf9bcPxlqOHvwFkDwzpybTqmnn3yKEJSI09E6wJQZBAA. Accessed 18 September 2021.

Buitrago Flórez F, Casallas R, Hernández M, Reyes A, Restrepo S & Danies G 2017. Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research*, 87(4):834–860. https://doi.org/10.3102/0034654317710096

Chalmers C 2018. Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17:93–100. https://doi.org/10.1016/j.ijcci.2018.06.005

Ching YH, Hsu YC & Baldwin S 2018. Developing computational thinking with educational technologies for young learners. *TechTrends*, 62:563–573. https://doi.org/10.1007/s11528-018-0292-7

Denning PJ 2017. Remaining trouble spots with computational thinking. *Communications of the ACM*, 60(6):33–39. https://doi.org/10.1145/2998438

Hsu TC, Chang SC & Hung YT 2018. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, 126:296–310. https://doi.org/10.1016/j.compedu.2018.07.004

Malik SI & Coldwell-Neilson J 2017. A model for teaching an introductory programming course using ADRI. *Education & Information Technologies*, 22:1089–1120. https://doi.org/10.1007/s10639-016-9474-0

Mao J, Ifenthaler D, Fujimoto T, Garavaglia A & Rossi PG 2019. National policies and educational technology: A synopsis of trends and perspectives from five countries. *TechTrends*, 63:284–293. https://doi.org/10.1007/s11528-019-00396-0

Motshekga A 2019. *Remarks at the media briefing on the occasion of the last post-Council Of Education Ministers' meeting*. Pretoria, South Africa, 8 March. Available at https://www.education.gov.za/Newsroom/Speeches/tabid/950/ctl/Details/mid/8127/ItemID/6002/Default.aspx. Accessed 30 October 2019.

Nardelli E 2019. Do we really need computational thinking? *Communications of the ACM*, 62(2):32–35. https://doi.org/10.1145/3231587

Papert S 1980. *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Rich PJ, Browning SF, Perkins M, Shoop T, Yoshikawa E & Belikov OM 2019. Coding in K-8: International trends in teaching elementary/primary computing. *TechTrends*, 63:311–329. https://doi.org/10.1007/s11528-018-0295-4

Shute V, Sun C & Asbell-Clarke J 2017. Demystifying computational thinking. *Educational Research Review*, 22:142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Strawhacker A & Bers MU 2019. What they learn when they learn coding: Investigating cognitive domains and computer programming knowledge in young children. *Educational Technology Research and Development*, 67:541–575. https://doi.org/10.1007/s11423-018-9622-x

Wing JM 2006. Computational thinking. *Communications of the ACM*, 49(3):33–35. Available at https://dl.acm.org/doi/fullHtml/10.1145/1118178.1118215?casa_token=6hfP6Mij2AAAAAAA:V7WYzhul07U8EKV3LS4WYtlGTMty_-nzyUKIOP6X5WgGtEKxkXQ5JKM74La_niky8d08agm3j1I7Emc. Accessed 16 September 2021.

Yadav A, Krist C, Good J & Nadire Caeli E 2018. Computational thinking in elementary classrooms: Measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, 28(4):371–400. https://doi.org/10.1080/08993408.2018.1560550

Zaharija G, Mladenović S & Boljat I 2013. Introducing basic programming concepts to elementary school children. *Procedia - Social and Behavioral Sciences*, 106:1576–1584. https://doi.org/10.1016/j.sbspro.2013.12.178