IJCSES

Volume 7, No: 1
Nov 2024
ISSN 2513-8359

International Journal of Computer Science Education in Schools

Editors

Dr Filiz Kalelioglu

Dr Yasemin Allsop

www.ijcses.org

International Journal of Computer Science Education in Schools

Nov 2024, Vol 7, No 1

DOI: 10.21585/ijcses.v7i1

Table of Contents

The perceptions of secondary school students regarding the learning of computer programming at Key Stage 4 in the revised curriculum in England: difficulties and prospects.

Almabrok Musa ALMDAHEM
a.almadahem@wlv.ac.uk
Faculty of Science and Engineering, School of Computer Science
University of Wolverhampton, England, UK

DOI: 10.21585/ijcses.v7i1.147

Abstract

A national curriculum for the study of computing became compulsory in English secondary schools in September 2014, replacing the study of information and communications technology with computer science (CS). This posed difficulties for teachers and students who did not have knowledge or experience of programming. This study was designed to investigate and gain a critical understanding of the teaching of computer programming (CP) at Key Stage 4 (KS4; year 10 - 11) of the CS curriculum, including assessing the impact of learning CP and students' perceptions of CS and their overall performance in the subject. Furthermore, the study investigated the measures to improve the teaching of CP and the factors that have an impact on the effective teaching of the CP curriculum. The study sample comprised 300 students. The findings indicate that the main difficulties the study found that the issues faced by students learning programming include a lack of time, the perceptions that it is a 'difficult' subject and students' insufficient understanding of programming. The findings also suggest that schools have made efforts to overcome these challenges and are willing to adopt programming as a subject and to help, encourage, develop and improve students' ability to learn programming; however, the results indicate that it is essential that schools address the shortage of teaching staff with specialised knowledge of CP. This study revealed that three factors can help to overcome the difficulties where the three factors are for students (perceptions towards learning and teaching programming, benefits, and support). The findings of this study will be useful for students who are learning programming in secondary schools.

Keywords: programming, National Curriculum, Key Stage 4(KS4), secondary school, difficulties, and learning.

1. Introduction

A 2012 report by the Royal Society advocated replacing the existing information and communications technology (ICT) curriculum in England with a wider-ranging subject to be known as 'computing' (The Royal Society, 2012). In 2014, the Department for Education (DfE) replaced England's national curriculum for ICT for secondary schools with a revised computing curriculum (Moller and Crick, 2018). Computing is now a compulsory part of the national curriculum for schools and provides important learning opportunities. The revised computing curriculum has three strands: computer science (CS); digital literacy (DL); and information technology (IT) (Lau, 2017) at Key Stage 4 (KS4; year 10 - 11) (see Figure 1). As computers are becoming an inseparable part of everyday life, the need and demand for computer programming (CP) is increasing rapidly. England was one of the first countries to take the initiative to integrate CS into its school curriculum (Passey, 2017). CS is the study of both software and hardware design, including principles of information processing and how digital systems work. In the CS element of the curriculum, students are taught the basic principles of programming, how digital systems work, and how to put this knowledge to use through programming. For many students, programming is regarded as one of the most challenging aspects of CS.

Several computing education researchers have sought to establish the causes of students' programming difficulties and have identified the lack of knowledge as one of the contributors (Sentance, Waite, and Kallia, 2019). Computer programming is becoming increasingly important to many societies around the world, and is a skill required by most educational institutions. However, the teaching of programming is not well developed in many secondary schools. Today, the teaching of programming is considered to be a priority in several countries; hence the interest in this domain and the extent to which research in the field is growing. The revised national curriculum for computing was introduced by the Department for Education (DfE) in England in September 2014 with the intention of providing students with the necessary skills and knowledge in this area of study (Larke, 2019). It replaced the

national curriculum for ICT in secondary schools (Moller and Crick, 2018). The 2014 revised curriculum included the application of mathematical skills, such as abstraction, decomposition (divide the problem to small parts to be manageable and easier to understand), logic, algorithms, and data representation. As computing is a key curriculum subject in all types of schools, this shift requires support for teachers to encourage new knowledge; teachers and students should therefore be given clear guidance on using computers successfully to support the teaching and learning of the subject (De Paula, Valente and Burn, 2014). This change brought with it a number of challenges. Prior to the introduction of the revised curriculum, ICT was often limited to the development of media, office-type software, and exploration of web-based resources (Woollard, 2017). The terms 'computer science and 'programming' are used in the revised curriculum; however, these words are not interchangeable, and these terms are defined in Chapter 2. According to the Royal Society (2019), there are 3,954 teachers of computing and 8,834 ICT teachers. RSA Oxford, Cambridge (OCR) found that there were 50,605 CS students in 2018. There are 24,323 schools in England, of which 3,448 are secondary schools (British Educational Suppliers Association; BESA, 2019).

1. What are KS4 students' perceptions of the learning of programming?

This paper investigated students' perceptions of the learning of programming. (see page 2)

2. Are there ways in which the teaching of CP can be enhanced? (see page 11)

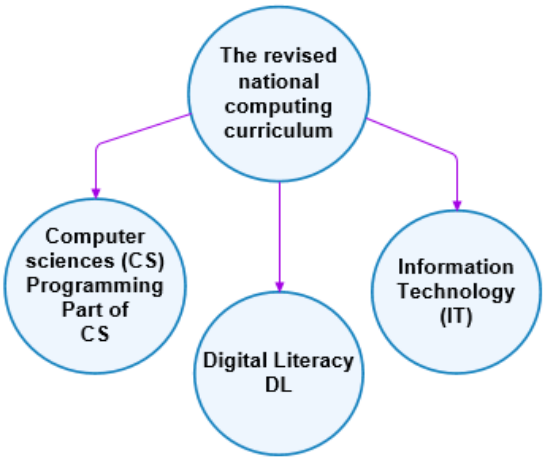

Figure 1. The national curriculum's computing programme of study (Lau, 2017, p.4).

2. Literature review

2.1 Significance of the study

This study addresses the way that the revised computing curriculum handles programming instruction, and it assesses the factors affecting effective implementation of the curriculum as well as the learning of programming. It also seeks to understand students' perceptions of this topic. Learning CS involves learning programming; therefore, it is important to develop students' fluency in this area of study. To become competent in programming, students need to be able to comprehend the concepts and use them well. Although this may seem to require extra time and effort, it is central to the learning of CS. Moreover, the original contribution of this study lies in the discussion related to the implementation of the revised CS curriculum in England's secondary schools. For a pedagogical model to be successful, it is important to examine its construction and effectiveness of application in secondary schools. This study contributes to the literature by considering students' opinions of such challenges for future improvement of CS/CP learning.

2.2 Students' perception

One of the difficulties faced by students when learning programming is the lack of lesson time. Students aged 14 years typically have one hour per week of computing lessons (The Royal Society, 2017). A recent survey showed that 30% of secondary schools reported a decrease in the total time allocated to teaching CS, while 22% saw an increase (Royal Society, 2017). As mentioned by 40% of the surveyed secondary schools that only provide one hour a week or fortnight for lessons in computing for 11–14-year-olds, teachers do not have sufficient time to ensure that the subject is adequately covered; therefore, enough time needs to be allocated in school timetables for covering the three strands of the computing curriculum (The Royal Society, 2017). A reduction in teaching hours would also make it more difficult for teachers to gain enough experience and confidence to teach programming (The Royal Society, 2017).

An additional issue that has an effect on students' learning is being taught by teachers who are not subject specialists in programming; in fact, when a specialist teacher is absent from class, students'

performance can decline, especially if there is no teacher with the same expertise to cover the absent teacher's role (Ost and Schiman, 2017). Some studies show that the numerous problems encountered in learning programming lead to the misunderstanding of the concepts of programming. For students, these issues may become obstacles to gaining programming skills, and these and other similar factors affect their progress in learning programming (Yukselturk and Altiok, 2017).

Another issue is students' mathematical skills. Duran (2016) tested the hypothesis that excellent mathematical skills yield excellent academic performance in CS; programming involves solving problems by applying mathematics and using computers for calculations. The ability to programme can simply be transferred to the performance of mathematical tasks. The development of the knowledge and understanding of programming will equip students with the creativity and skills to use a variety of new technologies.

2.3 Learning programming

People are already living in a world controlled by software, which is why it is so important for children to learn the basic elements of programming. For example, television is delivered over the internet; telephone calls are transmitted over software-controlled networks; people do not buy maps anymore but use the web; medical care is delivered online; and people buy their goods by shopping online. The next generation's world will be even more online and digital (Crow, 2014). Programming was once thought to be a task reserved for computer scientists, but in the twenty-first century, it is seen as a crucial and required talent that everyone should master (Shim, Kwon and Lee, 2016). Educational systems around the world are encouraging students to engage in programming activities and develop their programming skills (Scherer, Siddiq and Sánchez Viveros, 2019). High-quality teaching and learning of programming can help students to meet the digital challenges of the 21st century (Yildiz Durak, 2018). Programming is believed to help students succeed in other subjects in school, and it positively impacts on students' future employment prospects. It is a beneficial educational activity that helps students to develop and improve in other skills, such as problem-solving, and boost critical thinking and logical reasoning skills (Kalelioglu, and Gülbahar, 2014). Programming has a positive

effect on high school students' reasoning skills and self-efficacy problem-solving in mathematics. The learning of programming will also improve students' skills in other subjects and creativity and enhance collaboration. In fact, collaboration between students will considerably improve individual programming skills by reducing the frustration experienced by students and increasing their enjoyment and satisfaction in learning programming. Students will also be better prepared to collaborate as a group, for example, in pair programming. The retention of students in CS courses will also be improved (Li, Plaue and Kraemer, 2013).

In the future, students who have programming knowledge will be able to be innovative and solve problems more effectively, with fewer obstacles to impede their success. There is no doubt that the study of programming is beneficial to all students in their everyday life; the benefits also extend to positive impacts on processing, thinking, and communication (Jancheski, 2017). Sáez-López, Gonzalez and Cano (2016) argued that programming lessons can be beneficial to school students and that they should be engaged in programming, The value and success of applying visual programming from active methodologies education are highlighted by a grasp of computational ideas, project-based learning, active approach, usefulness, and commitment and motivation. This was confirmed by Laylaec (2019); there is no doubt that learning programming can be beneficial to students in the future, relating, for example, to improving employment and academic study opportunities. Taylor, Vasquez and Donehower, (2017) stated that as technology became more personalized, students should be given the opportunity to study programming, as this knowledge will equip them well for the future. Students who have programming knowledge and skills may be able to solve many of the problems of society by using their know-how of computer technologies and inventive notions in the future, encouraging students to learn motivation and understanding for programming by training and collaborative learning is important (Hayashi, Fukamachi and Komatsugawa, 2015). According to Psycharis and Kallia (2017) learning programming may also provide many benefits for students' cognitive skills which can be applied to a variety of subjects, this indicates that learning programming can help students develop skills that can be applied to other subjects such as mathematics, science, and engineering.

Programming is increasingly considered to be a significant skill in modern societies and, as Nager and Atkinson (2016) highlighted, it can lead to many employment opportunities, whilst it is a fundamental skill featured in the revised national curriculum for computing. Some research shows that the study of programming is becoming less prevalent, and recently the number of students choosing to study CS courses has also because of difficulties experienced in gaining CS skills (Azmi, Iahad and Ahmad,2015). Such difficulties include the fundamental concepts of programming, for example, construction loops, structure control and algorithms (Eltegani and Butgereit, 2015). However, one of the most difficult issues in education is how to keep students motivated, encouraged and interested in the learning (Eltegani and Butgereit, 2015). For students, learning programming is onerous because it requires considerable work, dedication, and training. The difficulties of learning programming are a cause for concern everywhere where this subject is needed (Vahldick, Mendes and Marcelino, 2014; Figueiredo and García-Peñalvo, 2018). These difficulties include, for example, lack of resources, such as textbooks for students; students' difficulties in understanding programming concepts; the lack of experience of the teacher giving programming lessons; students finding the lessons 'boring'; and insufficient teaching and learning time (Sentance and Csizmadia, 2017). On the other hand, motivation and interest are significant variables in the learning of programming. The lack of these attributes will push many students to give up CP; therefore, several studies have been carried out in an effort to improve students' motivation and interest in programming (Shim, Kwon and Lee,2016). Further study is still needed to uncover the different problems encountered in learning this subject; students frequently experience difficulties in grasping basic and essential concepts of programming, leading to disappointment and confusion (Galgouranas, and Xinogalos, 2018). Despite this increasing lack of interest in CS among students (Combéfis, Beresnevičius and Dagienė, 2016), the NCCE revealed in 2019 that it would open 23 new computing centres across England, providing assistance, including teaching support and resources, to teachers of secondary computing (Snowdon, 2019).

2.4 Student support

Support and motivation for learning programming in school are important for students. According to Kafai and Burke (2015), the use of educational games in the classroom is beginning to be seen as promising because of the evidence that such games can increase student performance and motivation. Motivation has a significant role in academic achievement: higher motivation can result in increased academic achievement. Because the learning of programming necessitates constant practice, maintaining students' motivation is of the highest importance. Students need stronger support to stimulate them to become engaged in learning activities as well as support for collaboration to motivate and improve different models of teamwork (Khaleel, Ashaari, Wook, and Ismail,2017; Santos, Gomes,and Mendes, 2010).

Teachers also have a large role in supporting their students; this is especially the case for teachers who have greater experience in programming and can support students by performing a variety of activities and demonstrating skills in the field of programming. According to Alsubaie (2016), teachers have a responsibility to develop appropriate instructional strategies to help schools achieve curriculum objectives, as well as developing suitable approaches to students' learning. Moreover, teachers should support students in improving the skills needed for success in all their courses. It is also important to enhance the learning environment of students to offer a world-class computing education (O'Kane, 2019). Providing effective support or guidance is the key to the improvement of students' performance (Yang, Hwang, Yang, and Hwang, 2015). It is essential to provide students with good programming tools as practice facilitates the study of programming (Kazimoglu, Kiernan, Bacon, and MacKinnon, 2012). There is a big role for teachers in supporting students, both from a motivational standpoint and from a pedagogical standpoint. It may be worth considering ways to track and boost students' motivation and self-confidence. Specifically, appropriate instructional and pedagogic techniques will increase students' motivation, self-confidence, and perceptions of competence, thus increasing their willingness to put in the effort necessary to learn how to program. Further, learning what influences programming favourably and unfavourably can help students overcome the inherent challenges of

learning programming by creating the ideal learning environment and pedagogical approaches. So, motivation is crucial in learning programming. As a result, any instructional strategy should include motivating student techniques; this is especially true in courses where a very active student attitude is fundamental (Gomes et al., 2018).

2.5 Difficulties in learning programming

It is known that many students have difficulty learning programming, especially its concepts; a number of studies have shown that students may not have sufficiently developed skills and the knowledge to start learning programming (Wang et al., 2017). The learning of programming can be considered as an iterative operation. In the beginning, the student is taught simple and basic information and where to apply it. Students need to grasp fundamental programming concepts (for example, repetition, sequence, condition, branch, variable and function) and learn the use of instructions and syntax for programming and tools (Moons and De Backer, 2013; Shim, Kwon and Lee, 2016). According to Yukselturk and Altiok (2017), some students can have trouble obtaining the requisite competencies while studying programming so that lessons become challenging. However, there is evidence in the literature that some of the major difficulties of learning programming are ineffective learning, lack of interest in programming and lack of motivation to study programming (Khaleel et al., 2017). Students encounter several problems, for example, misunderstanding of programming and lack of resources and time. Despite these views that programming is difficult to learn, Luxton-Reilly (2016) states that it is actually easy to learn and that, with little effort, almost anyone can learn programming; all that learners need to do is collectively shift their mindset and reach achievable goals. Thus, it is important to recognise and study new teaching methods that focus on students' learning and ameliorating difficulties, consequently resulting in students' active involvement in learning (Piteira, Costa and Aparicio, 2018).

3. Method

3.1 Research design

The methodology used in this study is based on a mixed-methods approach, which usually involves gathering, analysing, and integrating data collected from qualitative approaches, such as open-ended, and quantitative data from surveys. Data were collected from secondary school students through a questionnaire and then an analysis of the collected data was undertaken. The last stage was the interpretation and analysis of the data. Based on the nature of the sample used in this study, the questionnaire was created for KS4 secondary school students. The questionnaires designed for students included four factors: students' perception; students' learning; support; and difficulties.

3.2 Participants

The participants in this research were secondary school students. Responses for the questionnaire came 32 schools and it was proposed that all the Year 10 and 11 students involved in the fill-in were asked to complete questionnaires to ascertain their perceptions about the learning of programming. The aim and objectives of the research were explained, including the use of questionnaires and interviews. Altogether, a total of 300 students were selected to complete questionnaires and about 10 students were selected for participation in open-ended questions. There were both female and male students aged 15-16 years old, and the data was collected in secondary schools in the several schools in England.

3.3 Ethical approval

According to the UK Data Protection Act (1998/2018), anyone processing, obtaining, holding or disclosing personal data must comply with the data protection principles. Personal data include sensitive information such as factual information about and personal opinions of the individual. The participants of this study were informed that their personal data would be processed in accordance with the rights of data subjects and would be destroyed after the project. It was also explained that the data would be protected from unauthorised or unlawful processing and would not be transferred to another country. Moreover, the researcher secured the personal data of the participants by not

recording names on the questionnaires or using names in the presentation of the findings of the study. The researcher's role in the research process was explained to the students. The ethical guidelines of the British Educational Research Association (1992) were adhered to throughout this study. These guidelines emphasise respect for persons, knowledge, democratic values, and quality in educational research.

3.3.1. Ethical considerations of research with children

Dealing with children implies the need for a degree of the right to respect. It also underlines the importance of carefully protecting children's rights throughout the study process (Pillay, 2014). Researchers should ensure that children are not harmed in any way through their participation in research (Broström, 2006).

3.3.2. Obtaining consent

For the child participants who were under the age of 18 years old in this study, approval was obtained from parents or legal guardians. According to Heath et al. (2007), obtaining informed consent from parents / guardians of children is vital to the ethical research process. Since children are frequently less familiar with what research necessitates, they may initially wish to participate but later feel less keen as they get to know what is involved. As a result, consideration should be given to how children might be made to feel at ease with terminating their participation in the research if they so desire.

3.3.3. Confidentiality, anonymity, and safeguarding

In keeping with the topic of this research, stringent ethical measures were taken throughout the research process. The study's ethical approval was obtained by the Ethics Committee of the Faculty of Science and Engineering. Anonymity, confidentiality and safeguarding are ethical procedures designed to protect the privacy of human subjects while collecting and analysing. As part of this study, no children have been mentioned by name and the results from individual students cannot be attributed to a single school in the presented results.

3.3.4. Respect

This research gave all students the right to express their views about their own experiences in their life of study. In addition, participants in this research are seen as indispensable and worthy partners in research. The outcomes of the study were therefore achieved by promotion, protection, and respect of the rights of students are made intrinsic to every stage and level of research. The collection of participants' views and ideas (students' perceptions in this study) about a social phenomenon (the teaching of programming) seems to be a valuable way to generate credibility and gain trustworthiness and respect.

4. Data analysis

The quantitative analysis of the data collected from the survey questionnaires completed by 300 students who participated in the study. The analysis focused on students' perceptions of learning programming, as well as the challenges encountered in learning/teaching programming in secondary schools. To investigate and gain a critical understanding of the learning of programming in the KS4 computing curriculum, Spearman's rank-order correlation coefficients and multiple linear regressions were used to determine the following: for students, whether there was a relationship between difficulties experienced when learning programming and the perceptions of learning programming, benefits of learning programming, and the teaching support provided at school for programming.

4.1 Quantitative findings

Table 1. What are the perceptions of KS4 students of programming in the revised curriculum?

						Male (N = 160)				**Female (N = 140)**		
Factor (1) perceptions of teaching programming / Effect factors on teaching programming												
Q6	My positive perception of learning programming helps me study better.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M

												(SD)
	21 (13.1)	22 (13.8)	20 (12.5)	74 (46.3)	23 (14.4)	3.4 (1.3)	13 (9.3)	16 (11.4)	19 (13.6)	70 (50.0)	22 (15.7)	3.5 (1.2)
Q7	When I find all the necessary resources and good teachers at school, I am motivated to study programming.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	10 (6.3)	26 (16.3)	16 (10.0)	49 (30.6)	59 (36.9)	3.8 (1.3)	12 (8.6)	35 (25.0)	19 (13.6)	36 (25.7)	38 (27.1)	3.4 (1.3)
Q8	Competition in learning programming with my classmates pushes me to perform better.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	2 (1.3)	9 (5.6)	28 (17.5)	89 (55.6)	32 (20.0)	3.9 (0.8)	5 (3.6)	5 (3.6)	16 (11.4)	83 (59.3)	31 (22.1)	3.9 (0.9)
Q9	The pressure from a teacher and my classmates forces me to learn to programme better and work harder.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	3 (1.9)	6 (3.8)	21	70	60	4.1 (0.9)	3 (2.1)	2 (1.4)	12 (8.6)	73	50 (35.7)	4.2

			(13.1)	(43.8)	(37.5)					(52.1)		(0.8)
Q10	When my classmates do better, I am motivated to study harder to keep up											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	3 (1.9)	4 (2.5)	19 (11.9)	99 (61.9)	35 (21.9)	4.0 (0.8)	3 (2.1)	5 (3.6)	22 (15.7)	78 (55.7)	32 (22.9)	3.9 (0.8)
	Factor (2) Benefits of learning programming											
Q11	Learning programming in secondary school will enhance my confidence in this subject in the future.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	4 (2.5)	5 (3.1)	16 (10.0)	87 (54.4)	48 (30.0)	4.1 (0.9)	3 (2.1)	8 (5.7)	14 (10.0)	77 (55.0)	38 (27.1)	4.0 (0.9)
Q12	Learning programming in secondary school will improve my performance in other subjects											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	30 (18.8)	8 (5.0)	9 (5.6)	110 (68.8)	3 (1.9)	3.3 (1.2)	28 (20.0)	6 (4.3)	14 (10.6)	83 (59.3)	9 (6.4)	3.3 (1.3)

Q13	Learning programming in secondary school will make it easier to complete studies in this subject at university level.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	3 (1.9)	9 (5.6)	16 (10.0)	79 (49.4)	53 (33.1)	4.1 (0.9)	5 (3.6)	4 (2.9)	20 (14.3)	74 (52.9)	37 (26.4)	4.0 (0.9)
Q14	Learning programming in secondary school gives me a chance to continue learning to programme in the future.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	4 (2.5)	5 (3.1)	22 (13.8)	79 (49.4)	50 (31.3)	4.0 (0.9)	3 (2.1)	8 (5.7)	30 (21.4)	65 (46.4)	34 (24.3)	3.9 (0.9)
Factor (3) Difficulties experienced when learning programming												
Q15	Learning to programming at secondary school will motivate me to study it in the future.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	4 (2.5)	7 (4.4)	20 (12.5)	96 (60.0)	33 (20.6)	3.9 (0.9)	5 (3.6)	4 (2.9)	29 (20.7)	84 (60.0)	18 (12.9)	3.8 (0.8)
Q16	Learning programming is difficult.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	3	4	10	78	65	4.2	2	3	17	77	41	4.1

	(1.9)	(2.5)	(6.3)	(48.8)	(40.6)	(0.8)	(1.4)	(2.1)	(12.1)	(55.0)	(29.3)	(0.8)
Q17	Challenging programming exercises motivate me to work harder.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	2 (1.3)	19 (11.9)	14 (8.8)	73 (45.6)	52 (32.4)	4.0 (1.0)	1 (0.7)	17 (12.1)	10 (7.1)	73 (52.1)	39 (27.9)	3.9 (0.9)
Factor (4) Teaching support provided at school for programming												
Q18	Support for teaching computer programming is good in secondary school.											
	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)	S.D (%)	D (%)	N (%)	A (%)	S.A (%)	M (SD)
	0 (0.0)	5 (3.1)	24 (15.0)	109 (68.1)	22 (13.8)	3.9 (0.6)	0 (0.0)	10 (7.1)	20 (14.3)	97 (69.3)	13 (9.3)	3.8 (0.7)

Note. S.D = strongly disagree D disagree; N = neither disagree nor agree; S.A = strongly agree A= agree; M = mean; SD = standard deviation; % = percent

Table 2. Descriptive statistics and Cronbach's alpha analysis of the factors affecting students 'learning of programming.

	Factor	**Items**	**M**	**SD**	**Cronbach's alpha**
Students	Perception of learning programming	5	3.80	0.47	0.830
	Benefits of learning programming	4	3.82	0.54	0.780

Difficulties experienced when learning programming	3	3.99	0.66	0.704
Teaching support provided at school for programming	1	3.87	0.67	NA

Note: M = mean; SD = standard deviation; NA = not applicable.

To determine whether there was a relationship between difficulties experienced by students when learning programming and their perception of learning programming, benefits of learning programming, and teaching support provided at school for programming, Spearman's rank-order correlation tests and multiple linear regression were employed. According to the results of Spearman's rank-order correlation tests, there was a significantly negative relationship between the score for difficulties experienced when learning programming and those for perception of learning programming ($rs = -0.182$, $p = 0.002$), benefits of learning programming ($rs = -0.345$, $p < 0.001$), and teaching support provided at school for programming ($rs = -0.331$, $p < 0.001$).

Table 3. Spearman's rank-order correlation coefficients for the relationship between the score for challenges experienced by students when learning programming and those for the three factors of interest.

Factor of interest	**Correlation coefficient (*p*-value)**
Perception of learning programming	-0.182 (0.002)
Benefits of learning programming	-0.345 (< 0.001)
Teaching support provided at school for programming	-0.331 (< 0.001)

The results of the multiple linear regression (Table 4) show that the predictor, the score for perception of learning programming, contributed significantly to the model ($t(96) = -2.728$, $p = 0.007$). The relationship between the score for difficulties experienced when learning programming and that for

perception of learning programming was significantly negative (B = -0.199, SE = 0.073). That is, students perceiving learning programming more positively experienced less difficulty when learning programming. Based on these results, $H0_{1s}$ (no relationship between the score for difficulties experienced when learning programming and that for perception of learning programming) was rejected as the regression coefficient was significantly negative, with a *p*-value of < 0.05. Therefore, it was concluded that the more positive the students' perception of learning programming was, the less the difficulty they experienced when learning programming. The predictor, the score for benefits of learning programming, contributed statistically significantly to the model (t (96) = -6.472, $p < 0.001$). There was a significantly negative relationship between the score for difficulties experienced when learning programming and that for benefits of learning programming (B = -0.426, SE = 0.066). That is, students perceiving greater benefits of learning programming experienced less difficulty when learning programming. Based on these results, $H0_{2s}$ (no relationship between the score for difficulties experienced when learning programming and that for benefits of learning programming) was rejected as the regression coefficient was significantly negative, with a *p*-value < 0.05. Therefore, it was concluded that the greater the students' perception of benefits of learning programming, the less difficulty they experienced when learning programming. The predictor, the score for teaching support provided at school for programming, contributed statistically significantly to the model ($t(96) = -5.234$, $p < 0.001$). The relationship between the score for difficulties experienced when learning programming and that for teaching support provided at school for programming was negative (B = -0.264, SE = 0.050). That is, students who had a higher perception of teaching support provided at school for programming experienced less difficulty when learning programming. Based on these results, $H0_{3s}$ (no relationship between the score for difficulties experienced when learning programming and that for teaching support provided at school for programming) was rejected as the regression coefficient was negative, with a *p*-value value of < 0.05. Therefore, it was concluded that the greater the students' perception of teaching support provided at school for programming, the less difficulty they experienced when learning programming.

Table 1. Results of the multiple linear regression for determining the relationships between the score for difficulties experienced by students when learning programming and those for three factors of interest.

Factor	**B**	**SE**	**t**	**p**	**VIF**
Constant	5.026	0.098	51.047	< 0.001	
Perception of learning programming	-0.199	0.073	-2.728	0.007	1.189
Benefits of learning programming	-0.426	0.066	-6.472	< 0.001	1.267
Teaching support provided at school for programming	-0.264	0.050	-5.234	< 0.001	1.152

Note B = parameter estimate, SE = standard error, *t* = *t*-statistic, *p* = *p*-value, and VIF = variance inflation factor.

Summary of the findings of the quantitative analysis aimed to investigate students' perceptions of learning and programming. The results indicate that students had a positive perceptions of learning programming and perceived it as beneficial. However, they experienced high levels of difficulty when learning programming but believed that teaching support provided at school for studying programming was good. To determine whether there was a relationship between difficulties experienced by students when learning programming and their perceptions of learning programming, benefits of learning programming, and teaching support provided at school for programming, Spearman's rank-order correlation tests and multiple linear regression were employed. According to the results of Spearman's rank-order correlation tests, there was a significantly negative relationship between the score for difficulties experienced when learning programming and those for perceptions of learning programming, benefits of learning programming, and teaching support provided at school for programming. Similarly, based on the results of the multiple linear regression, there was a statistically significantly negative relationship between the score for difficulties experienced when learning programming and those for perceptions of learning programming, benefits of learning programming, and teaching support provided at school for programming.

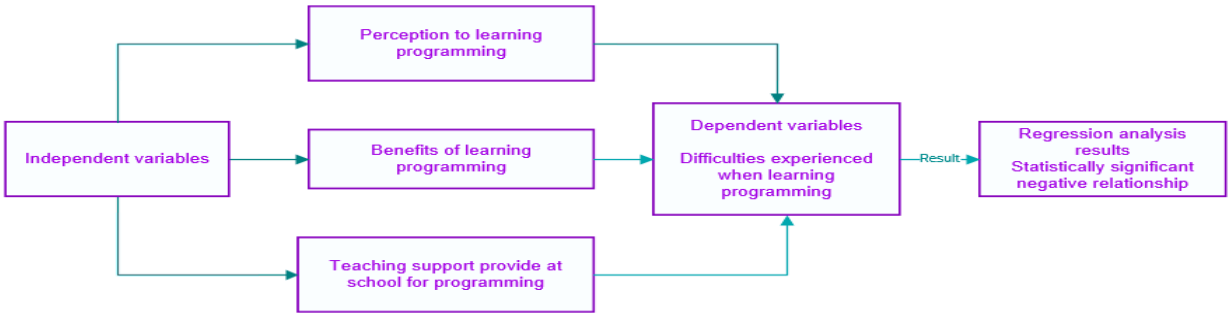

Figure 2. Regression analysis of data for students.

The results indicated that students with a more positive perception towards learning programming and perceived greater benefits of learning programming and greater teaching support provided at school for programming experienced less difficulty when learning programming (Figure 2). Cronbach's alpha was computed for items under each factor to determine the reliability of the construct. The results of Cronbach's alpha analysis indicated a high reliability of the construct for students. The quantitative results indicated that students learning programming difficulties lie in a lack of time, lack of content programming lack of experiences lack of programming knowledge, lack of motivation the perception that it is a 'difficult' subject, and students' insufficient understanding of programming. Students believed that when they perceived more support, motivation, good resources, and skilful programming teachers, and more activities, they would be more interested and motivated to learn programming. The quantitative results were significantly negative, which suggests that for students the difficulties in learning programming would be overcome by an increase in support, benefits, and perceptions.

4.2 Qualitative findings

This section discussed the responses garnered through open-ended surveys of students. It provides important explanations for the learning of programming. The results of the qualitative analysis of research question 1 confirmed that students, both males and females, received support from their schools to study programming and that programming clearly helped students in their learning of other subjects. The responses to the open-ended questions provided some evidence that the learning of programming enhanced future opportunities for higher education and careers. There were no gender

differences in these responses. In addition, the findings for research question 1 show that most of the respondents noted and confirmed the benefits of introducing programming into the Key Stage 4 curriculum on the basis that programming is useful and that it is an important skill necessary for students in the future. The responses to research question 2 reveal that the development of students' confidence in studying CP would facilitate the learning of complicated topics in communication processes and practices of programming. Having presented the analysis of the relevant data of this study.

5. Discussion

5.1 Research question 1: What are KS4 students' and teachers' perceptions of the learning of programming?

5.1.1 Students' perceptions

In order to discuss the data generated from this research question, this section starts with presenting the students' perceptions. One of the main knowledge areas within the CS curriculum is programming, which includes algorithms, concepts, patterns, programming paradigms and technologies (Halim and Phon, 2020). The results of the quantitative analysis indicate that when students are provided by suitable classroom environment, resources, and skilful programming teachers, they will be interested and motivated to learn programming. In addition, students believe that learning programming enhances their confidence in future studies and improves their achievements in other subjects, and they feel that practical learning support leads them to perform better. However, many students believe that they need to learn programming, but they realise that the learning of programming is not an easy task. Practical support is a significant component of programming courses, and it is an important process for students to develop their skills. 'Practice is considered an important step in grasping the precise concepts of computer programming for novices' (Malik, 2016, p.1).

Programming is often actively linked to learning in other subjects such as mathematics, science, and technology (Otterborn, Schönborn and Hultén, (2020). Programming helps students acquire skills that are prerequisites for success in other subjects, including problem-solving, critical thinking, creativity,

collaboration, mathematical thinking, and reasoning (Psycharis and Kallia, 2017; Tsai, 2019; Partovi, 2020). Learning programming in secondary schools affects students both in the moment and in the future. Saez-Lopez et al. (2020) reported that programming knowledge offers advantages and benefits related to various fields by providing the skills that stimulate motivation and digital competence. LópezLeiva et al. (2022) noted that students enjoyed using programming while applying mathematics to develop images and videos that they chose and created through programming. The findings of this study indicate that students considered that programming can help them in their educational career. In the context of this study, the majority of students believe that learning programming is a useful skill that will help them later in life. In addition, some students indicated that the learning of programming is a vital skill that all students must acquire. However, some students are not interested to learn programming, even though they were aware of the benefits. Thus, although future plans can be a contributing factor to choosing to learn CS, it is not the only factor to consider.

The revised computing curriculum was developed to provide young people with the computing knowledge, understanding, and foundational skills they require now and will require in the future (Dredge, 2014). The data obtained from the survey and the open-ended questions in this study showed that students and teachers both believe that programming education will help students in other subjects in school. However, some students may not be interested in the future study of programming at college or university level, or they may only take a short course after graduation from school and get a job, while other students who like programming in secondary school may not want to continue with it at university and instead, they tend to study other subjects.

In this study, students acknowledged that a good teacher is one of the significant elements of teaching programming. The teacher has a crucial role in the education process in class and school. Teachers are the primary source of knowledge and that can positively impact on the students' achievement. Some of the students indicated that the absence of a good teacher affects their learning of programming, although it is likely that they were referring to a direct effect through their teacher's absence or illness. This study indicates that in the event that there is no subject teacher, it is important to supply the class with a dedicated teacher for CS or programming. The study also provides evidence that experienced

teachers will have a direct effect on students' success (Gage et al., 2018). Moreover, the absence of a teacher specialising in the field has an effect on students' learning, especially when the replacement does not meet the same requirement (Ost and Schiman, 2017). It is important to provide support and professional development opportunities for those who are currently teaching computing in schools (Moller and Powell, 2019). This is a major reason for guiding and helping students to improve in their study of programming. The results of this study showed that students with a more positive attitude to learning programming experienced fewer difficulties when learning the subject. This study also showed that a large percentage of students did not have a computer at home. The absence of a home computer may lead to students' lack of experience, confidence, and time for learning programming in their school. The results of this study showed that the majority of students do not own a computer at home, which was perceived to have a negative impact on their skills at programming. This finding is consistent with that of Fairlie and Robinson (2013) in that when students do have access to a home computer, they can gain experience, confidence, and the time to devote to programming. Advantages of home computers include helping students with their learning, students' increased desire to create resources and artefacts, development of more skills, improvement of their existing skills, greater experience, and increased flexibility in the times when they can use computers (Fairlie, 2012).

However, home computer use also has disadvantages, including the considerable amount of time spent playing games which leads to students having no energy or time for their studies. (Fairlie and London, 2012). Some students noted that they were not confident in learning programming and that this had an impact on their interest and motivation to learn the subject. This was also explained by Shim, Kwon and Lee (2016) who stated that many students see the field of programming as a difficult subject and students feel disappointed when they do not make the progress that they believe they should. Consequently, programming difficulty had an effect on students' decisions either to choose CS or not. Studying programming needs a significant amount of knowledge, skills, time, and practice, which does not inspire students who are looking to study CS as an option at school (Benjamin, 2017). Therefore, several scholars conclude that programming is a complicated process and there are challenges and problems in teaching and learning the subject (for example, Prasad and Chaudhary,

2021). However, maybe a lack of programming capacity is not the only reason why some students decide not to select studying programming, and there may be other reasons such as a lack of interest or motivation. Other important factors are students' poor grasp of programming content and lack of experience, which can make the learning of programming difficult. This is consistent with the fact that the study of programming requires inspiration, knowledge, ability, skills, time, and practice (Benjamin, 2017).

Another issue that makes programming difficult is mathematics. Research conducted by Mozelius, Ulfenborg and Persson (2019), showed that the lack of knowledge and mathematical skills makes programming a difficult subject; however, they believe that programming should have a positive effect on students' mathematical skills. This study concurred with Duran (2016) who tested the hypothesis that excellent mathematical skills yield excellent academic performance in CS, as programming is solving problems by applying mathematics. Programming is an important skill necessary for mathematics and sometimes programming failure rates can be, partially, attributed to a lack of mathematical capacity.

5.1.2 Are there ways in which the teaching of CP can be enhanced?

The result of this study shows that students believe that collaborative programming is important and an effective approach to learning. According to Bravo, Duque and Gallardo (2013), collaborative programming increases confidence and enhances the value of learning programming. Collaborative interactions in learning programming, such as pair programming, can develop more positive feelings and experiences than individual programming (Cal and Can, 2020). Identification of the factors that affect the achievements and confidence of students using pair programming can enable teachers and curriculum developers to make better decisions on the use of this approach in secondary school programming courses (Cal and Can, 2020). It should be noted, however, that collaborative learning is a complex method that entails the co-creation of knowledge (Tsan et al., 2021) and that further research is required to determine the impact of such pedagogic approaches. Demir and Seferoglu (2021), for example, noted that only a small number of studies have experimentally demonstrated that pair

programming is effective. The issue of on-going professional development was mentioned by some of the teachers, including more support with different pedagogical approaches that could help to stimulate students in the classroom environment.

The use of games was mentioned as a possibility. Some studies such as Papadakis (2020) mentioned that the game development and programming environment approach has a positive effect on students' motivation and achievement of basic programming skills in CS lessons. Moreover, since robots are real tools that assist students to understand the concepts of programming, the use of programme robots will enable students to enter a potentially fun and appealing learning environment (Alalawi and Said, 2020). Research conducted by Dlab et al. (2020) showed that the use of modern learning methods and appropriate digital content and tools, including games, is more effective for achieving teaching and learning goals. This research confirmed that the learning of programming through the use of games promotes good programming practices and enables students to understand the concepts of programming.

Several studies have focused on the issue of students' motivation for learning programming (for example, Zarei et al., 2020). High-performance computing artefacts provide students with opportunities to increase their understanding and improve their learning of CS (Mwasaga and Joy, 2020). Learning through educational games can contribute positively to the learning outcomes and increase the students' motivation in the learning programming (Mathew, Malik and Tawafak, 2019). Motivation plays a significant role in learning programming; it assists the students in learning the basic concepts of programming. Moreover, these games enhance competition and collaboration between students. These results broaden the knowledge and help to overcome the difficulties faced by students in the learning of programming in secondary schools and could shed light on future policymaking in relation to curriculum development for secondary schools.

6. Recommendations

Recommendation regarding the perceptions of students of programming

1) This study recommends that schools should motivate and support students to give them confidence in themselves to succeed in this field.

2) Schools should consider increasing the teaching time allocated to programming lessons by providing more extracurricular activities, for example, after-school programming clubs.

3) Schools should provide appropriate technical resources to support the teaching of programming.

4) Collaboration is important for students; it is a significant technique for developing higher-quality learning and is recommended as a pedagogical aid for CS/programming teachers (this study suggests pair programming as an important support for students).

5) Schools should consider providing high-performance computing artefacts (tools) to increase students' understanding, and improve their learning of CS.

6) In the case of students who do not have a home computer, schools should cooperate with parents to provide a computer or laptop to support and assist them in learning programming or any other subject.

References

Alalawi, W.A. and Said, M.N.H.B.M. (2020) Using Lego Mindstorms robotics programming in enhancing computational thinking among middle school in Saudi Arabia. *International Journal of Psychosocial Rehabilitation*, 24(05), pp. 5367–5373.

Alsubaie, M. A. (2016). Curriculum development: Teacher involvement in curriculum development. *Journal of Education and Practice*, *7*(9), 106-107.

Azmi, S., Iahad, N.A. and Ahmad, N. (2015) Gamification in online collaborative learning for programming courses: A literature review. *ARPN Journal of Engineering and Applied Sciences*, 10(23), pp. 1-3.

Benjamin Wohl (2017) *Coding the curriculum: new computer science GCSE fails to make the grade*[online][Accessed 5 January 2019]. available at: Coding the curriculum: new computer science GCSE fails to make the grade | The Independent | The Independent

Bravo, C., Duque, R. and Gallardo, J. (2013) A groupware system to support collaborative programming: Design and experiences. *Journal of Systems and Software*, 86(7), pp. 1759–1771.

Broström, S. (2006) Children's perspectives on their childhood experiences. Nordic childhoods and early education: Philosophy, research, policy, and practice in Denmark, Finland, Iceland, Norway, and Sweden, pp. 223-255.

Cal, H. and Can, G. (2020) The Influence of Pair Programming on Secondary School Students' Confidence and Achievement in Computer Programming. *Trakya Eğitim Dergisi,* 10(1), pp. 221-237.

Combéfis, S., Beresnevičius, G. and Dagienė, V. (2016) Learning programming through games and contests: Overview, characterisation and discussion. *Olympiads in Informatics*, 10(1), pp. 39–60.

Crow, D. (2014) Why every child should learn to code [online]? [Accessed 12 September 2018]. Available at: https://www.theguardian.com/technology/2014/feb/07/year-of-code-dan -crow-songkick

Demir, O. and Seferoglu, S.S. (2021) A comparison of solo and pair programming in terms of flow experience, coding quality, and coding achievement. *Journal of Educational Computing Research*, 58(8), pp. 1448–1466.

Dlab, M.H., Hoic-Bozic, N., Mezak, J. and Zunic, M., (2020) Supporting Croatian Primary School Teachers in Designing Game Based Learning Activities: A Case Study Martina Holenko Dlab , Natasa Hoic-Bozic , Jasminka Mezak , and Marina Zunic University of Rijeka, Department of Informatics, Rijeka, Croatia University of Rijeka, Faculty of Teacher Education. *ECGBL 2020 14th European Conference on Game-Based Learning (p. 125).* Academic Conferences limited.

Dredge, S. (2014) *Coding at school: A parent's guide to England's new computing curriculum. The Guardian* [online]. [Accessed 16 November 2017]. Available at: https://www.theguardian.com/technology/2014/sep/04/coding-school-computing-children-programming

Duran, I. L. (2016). The role of mathematics background in the performance of BSCS students in computer programming subject. *International Journal of Multidisciplinary Research and Modern Education (IJMRME)*, *2*(1), 147-50.

Eltegani, N. and Butgereit, L. (2015) Attributes of students' engagement in fundamental programming learning. *International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering* (ICCNEEE). IEEE, pp. 101–106.

Fairlie, R. W. (2012). The effects of home access to technology on computer skills: Evidence from a field experiment. *Information Economics and Policy*, *24*(3-4), 243-253.

Fairlie, R. W., & London, R. A. (2012). The effects of home computers on educational outcomes: Evidence from a field experiment with community college students. *The Economic Journal*, *122*(561), 727-753.

Fairlie, R. W., & Robinson, J. (2013). Experimental evidence on the effects of home computers on academic achievement among schoolchildren. *American Economic Journal: Applied Economics*, *5*(3), 211-40.

Figueiredo, J. and García-Peñalvo, F.J. (2018) Building skills in introductory programming. In: *Proceedings of the Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality*, pp. 46–50.

Gage, N. A., Scott, T., Hirn, R., & MacSuga-Gage, A. S. (2018). The relationship between teachers' implementation of classroom management practices and student behavior in elementary school. *Behavioral disorders*, *43*(2), 302-315.

Galgouranas, S. and Xinogalos, S. (2018) JAVANT-GARDE: A cross-platform serious game for an introduction to programming with Java. *Simulation & Gaming*, 49(6), pp. 751-767.

Gomes, A., Ke, W., Lam, C.T., Marcelino, M.J. and Mendes, A. (2018) Student motivation towards learning to program. In *2018 IEEE Frontiers in Education Conference (FIE)* (pp. 1-8).

Jancheski, M. (2017) Improving teaching and learning computer programming in schools through educational software. *Olympiads in Informatics*, 11, pp. 55–75.

Halim, N. F. A., & Phon, D. N. E. (2020, Februar). Mobile Learning Application Impact Towards Student Performance in Programming Subject. In *IOP Conference Series: Materials Science and Engineering* (Vol. 769, No. 1, p. 012056). IOP Publishing.

Hayashi, Y., Fukamachi, K.I. and Komatsugawa, H. (2015) Collaborative learning in computer programming courses that adopted the flipped classroom. In: *International Conference on Learning and Teaching in Computing and Engineering*. IEEE, pp. 209–212.

Heath, S., Charles, V., Crow, G. and Wiles, R., (2007) Informed consent, gatekeepers and go-betweens: negotiating consent in child and youth-orientated institutions. *British Educational Research Journal*, 33(3), pp. 403-417.

Kafai, Y.B. and Burke, Q. (2015). Constructionist gaming: Understanding the benefits of making games for learning. *Educational psychologist*, *50*(4), 313-334. *https://doi.org/10.1080/00461520.2015.1124022.*

Kalelioglu, F., & Gülbahar, Y. (2014). The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective. *Informatics in Education*, *13*(1), 33-50.

Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning programming at the computational thinking level via digital game-play. *Procedia Computer Science*, *9*, 522-531.

Khaleel, F. L., Ashaari, N. S., Tengku, T. S. M., & Ismail, A. (2017). Programming learning requirements based on multi perspectives. *International Journal of Electrical and Computer Engineering*, *7*(3), 1299.

Larke, L.R. (2019) Agentic neglect: Teachers as gatekeepers of England's national computing curriculum. *British Journal of Educational Technology*, 50(3), pp. 1137-1150.

Lau, W. (2017). Teaching Computing in Secondary Schools: A Practical Handbook. Routledge.

laylaec (2019) Advantages and disadvantages of programming education [online]. [Accessed: 2 August 2019]. Available at: Advantages and disadvantages of programming education (laylaec.com)

Laylaec (2019) *Advantages and disadvantages of programming education* [online]. [Accessed: 2 August 2019]. Available at: https://laylaec.com/2019/04/29/advantages-and-disadvantages-of-programming-education/

Li, Z., Plaue, C., & Kraemer, E. (2013, May). A spirit of camaraderie: The impact of pair programming on retention. In *2013 26th International Conference on Software Engineering Education and Training (CSEE&T)* (pp. 209-218). IEEE.

LópezLeiva, C.A., Noriega, G., Celedón-Pattichis, S. and Pattichis, M.S. (2022) From students to cofacilitators: Latinx students' experiences in mathematics and computer programming. Teachers College Record, 124(5), pp. 146-165.

Luxton-Reilly, A. (2016, July). Learning to program is easy. In Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (pp. 284-289).

Malik, S.I., (2016) Enhancing practice and achievement in introductory programming using an ADRI editor. In: *2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 32-39. IEEE.

Mathew, R., Malik, S.I. and Tawafak, R.M. (2019) Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. *Informatics in education*, 18(2), pp. 359-373.

Moller, F., & Crick, T. (2018). A university-based model for supporting computer science curriculum. Reform Journal of Computers in Education, 5(4), 415-434.

Moller, F., & Powell, S. (2019, January). Teaching Computing via a School Placement. In *Proceedings of the 3rd Conference on Computing Education Practice* (pp. 1-4).

Mozelius, P., Ulfenborg, M., & Persson, N. (2019). Teacher attitudes towards the integration of programming in middle school mathematics. *INTED 2019*, 701-706.

Moons, J., & De Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education*, *60*(1), 368-384.

Mwasaga, N.M. and Joy, M., (2020) Using high-performance computing artifacts as a learning intervention: A systematic literature review. *Proceedings of the 2nd International Conference on Intelligent and Innovative Computing Applications.* pp. 1–10.

Nager, A. and Atkinson, R.D. (2016) *The case for improving US computer science education* [online]. [Accessed: 12 January 2019]. Available at: https://www2.itif.org/2016-computer-science-education.pdf

Nikiforos, S., Kontomaris, C., & Chorianopoulos, K. (2013). MIT scratch: A powerful tool for improving teaching of programming. *Conferrence on Informatics in Education*, 1-5.

O'Kane, L. (2019) *Computing mastery for primary schools: Achieving computing mastery* [online]. Accessed: 2 May 2019]. Available at: computing mastery Archives - iCompute (icompute-uk.com)

Ost, B., & Schiman, J. C. (2017). Workload and teacher absence. *Economics of Education Review, 57,* 20-30.

Otterborn, A., Schönborn, K.J. and Hultén, M. (2020) Investigating preschool educators' implementation of computer programming in their teaching practice. *Early Childhood Education Journal,* 48(3), pp. 253-262.

Papadakis, S. (2020) Evaluating a game-development approach to teach introductory programming concepts in secondary education. *International Journal of Technology Enhanced Learning*, 12(2), pp. 127–145.

Passey, D. (2017). Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies, 22*(2), 421-443.

Pillay, J., (2014) Ethical considerations in educational research involving children: Implications for educational researchers in South Africa. *South African Journal of Childhood Education,* 4(2), pp. 194-212.

Piteira, M., Costa, C., & Aparicio, M. (2018). Computer programming learning: how to apply gamification on online courses?. *Computer programming learning: how to apply gamification on online courses?*, (2).

Prasad, A., & Chaudhary, K. (2021). Interactive Animation and Affective Teaching and Learning in Programming Courses. In *Advances in Computer, Communication and Computational Sciences* (pp. 613-623). Springer, Singapore.

Psycharis, S. and Kallia, M. (2017) The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, 45(5), pp. 583–602.

The Royal Society (2012) *Shutdown or restart? The way forward for computing in UK schools* [online]. [Accessed 19 April 2017]. Available at: https://royalsociety.org/-/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

Royal Society, *After the reboot: computing education in UK schools* [Online], (2017). [Accessed 12 September 2018]. Available at: <https://royalsociety.org/-/media/policy/projects/computing-education/computing-education-report.pdf/>

Sáez-López, J.M., Román-González, M. and Vázquez-Cano, E. (2016) Visual programming languages integrated across the curriculum in elementary school: A two-year case study using "Scratch" in five schools. *Computers & Education*, 97, pp. 129–141.

Sáez-López, J.M., del Olmo-Muñoz, J., González-Calero, J.A. and Cózar-Gutiérrez, R. (2020) Exploring the Effect of Training in Visual Block Programming for Preservice Teachers. *Multimodal Technologies and Interaction*, 4(3), pp. 2-10.

Santos, Á., Gomes, A., & Mendes, A. J. (2010). Integrating new technologies and existing tools to promote programming learning. *Algorithms*, *3*(2), 183-196.

Scherer, R., Siddiq, F. and Sánchez Viveros, B. (2019) The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. Journal of Educational Psychology, 111(5), pp. 764–792.

Sentance, S. and Csizmadia, A. (2017) Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*, 22(2), pp. 469–495.

Sentance, S. (2019). Moving to mainstream: developing computing for all. In *Proceedings of the 14th Workshop in Primary and Secondary Computing Education* (pp. 1-2).

Shim, J., Kwon, D., & Lee, W. (2016). The effects of a robot game environment on computer programming education for elementary school students. *IEEE Transactions on Education*, *60*(2), 164-172.

Snowdon, K. (2019) *First 23 computing hubs in England announced* [online]. [Accessed 12 October 2019]. Available at: https://schoolsweek.co.uk/computing-hubs-england-announced-ncce/

Taylor, M.S., Vasquez, E. and Donehower, C., (2017) Computer programming with early elementary students with Down syndrome. *Journal of Special Education Technology*, 32(3), pp. 149–159.

Tsai, C. Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior*, *95*, 224-232.

Tsan, J., Vandenberg, J., Zakaria, Z., Boulden, D.C., Lynch, C., Wiebe, E. and Boyer, K.E. (2021) Collaborative Dialogue and Types of Conflict: An Analysis of Pair Programming Interactions between Upper Elementary Students. In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, pp. 1184-1190.

Wang, X. M., Hwang, G. J., Liang, Z. Y., & Wang, H. Y. (2017). Enhancing students' computer programming performances, critical thinking awareness and attitudes towards programming: An online peer-assessment attempt. *Journal of Educational Technology & Society*, *20*(4), 58-68.

Woollard, J. (2017) Computer studies, information technology, ICT and now computing. In *Debates in Computing and ict Education*, pp. 14-26. Routledge.

Vahldick, A., Mendes, A.J. and Marcelino, M.J. (2014) A review of games designed to improve introductory computer programming competencies. In *2014 IEEE frontiers in education conference (FIE) proceedings* (pp. 1-7).

Yang, T.C., Hwang, G.J., Yang, S.J. and Hwang, G.H. (2015) A two-tier test-based approach to improving students' computer-programming skills in a web-based learning environment. *Journal of Educational Technology & Society*, 18(1), pp. 198–210

Yildiz Durak, H. (2018). Digital story design activities used for teaching programming effect on learning of programming concepts, programming self-efficacy, and participation and analysis of student experiences. *Journal of Computer Assisted Learning*, *34*(6), 740-752.

Yukselturk, E., & Altiok, S. (2017). An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology*, *48*(3), 789-801.

Zarei, A., Mohd-Yusof, K., Kolivand, H., Ahmadi, A. and Al-Hamar, Y. (2020) A new game-based strategy for enhancing youth programming skills. *Journal of Internet Technology*, 21(5), pp. 1289–1301.

International Journal of Computer Science Education in Schools, November 2024, Vol. 7, No. 1
ISSN 2513-8359

Investigating the Impact of Introducing Pair Programming to Primary Computing Education on Female Pupils' Attitudes Towards Computing

Katharine CHILDS
Raspberry Pi Foundation, Cambridge

Sue SENTANCE
Raspberry Pi Computing Education Research Centre,
University of Cambridge, Cambridge

DOI: 10.21585/ijcses.v7i1.218

Abstract

Gender balance in computing education is a decades-old issue that has been the focus of much previous research. In K-12, the introduction of mandatory computing education goes some way to giving all learners the opportunity to engage with computing throughout school, but a gender imbalance still persists when computer science becomes an elective subject. The study described in this paper investigates whether introducing pair programming would make a difference to primary-aged girls' attitudes to computing and intent to study the subject in the future. A randomised controlled trial (RCT) was designed and implemented around a 12-week intervention with 785 female pupils between the ages of 8 and 10 years, alongside a qualitative evaluation investigating teachers' and pupils' experience of the interventions and the development of materials and teacher preparation resources. The results of the RCT showed no statistically significant changes in student attitudes towards computing or intent to study further, although the qualitative data indicated that both teachers and pupils found the interventions engaging and enjoyable. Themes emerging from the qualitative data point to the importance of collaboration in supporting a development in pupil confidence. Overall, these results emphasise the societal and systemic barriers around computer science and technology engagement across genders that persist despite many initiatives being implemented over many years.

Keywords

K-12 computing education, pair programming, gender balance

1. Introduction

There is a decades-old problem with gender imbalance in computing education, at all levels of schooling and university, with an equally long history of research into the reasons why this is the case (Butler, 2000). Global statistics show that gender imbalances persist into the technology workforce, particularly in digital transformation roles such as data and cloud computing (Global Education Monitoring Report Team, 2024). Some evidence has shown that the gender stereotypes around computing affecting choices on further study develop early (Eccles, 2015), while there is less research available on specific approaches that can be used to address these stereotypes. Universal access to inclusive quality education offers a route to eliminating gender and wealth disparities (United Nations Development Programme (UNDP), n.d.) so in theory, mandatory computing education at primary and lower secondary phases of education provides a leveller to gender balance.

In school, girls do well in computing. An analysis of recent examination results in England demonstrates that girls are more likely to achieve higher grades than boys in formal computer science (CS) education (Kemp et al., 2019). However, girls may be less likely to be encouraged to continue with computing education or to take it in the first place (Cheryan et al., 2009) and amongst students who have chosen to study GCSE Computer Science, girls are less likely to aspire to be a computer scientist compared with boys (Hamer et al., 2023). Other studies have identified gender differences between learners in their attitudes towards computing. Male students are generally more confident using computers (Beyer et al., 2003) because they have more access and exposure to computers at home (Varma, 2009) and these gender disparities affect students' achievement in computing from as young as ten years old (Tsan et al., 2016).

This paper discusses the design, implementation, and evaluation of an intervention designed to investigate whether using pair programming – as part of primary computing education – could reduce barriers to female pupils' uptake of computing. The context for these studies is England, which has mandatory computing in Grades K-8 (primary and lower secondary education) with elective qualifications in CS offered for pupils at Grades 9-12 (upper secondary). Mandatory computing

education ensures that computing is offered to all children but when it becomes elective and young people choose a reduced number of subjects, educational statistics from 2022 report that only 21% of pupils taking CS at age 15/16 were female, and 15% at age 17/18. Therefore, this study was motivated by the need to maintain girls' interest during the mandatory stage of education.

2. Related work

2.1 Gender balance in computing

Work to research gender balance in computing is by no means new. A review conducted of gender balance research in computing from the 1980s and 1990s suggested that boys may have access to more role models in computing, may receive more encouragement to pursue the subject, and that software may be developed with a bias towards interests traditionally considered to be male (Butler, 2000). Given that this review dates from over two decades ago, gender balance in computing is an issue we've been trying to address for many years.

Initial studies framed the low number of female students as a puzzling phenomenon and used an abductive approach to create a metaphor to describe the imbalance. An example of this is the "shrinking pipeline" metaphor which illustrates the decreasing number of girls studying computing at each stage of education (Camp, 1997). Research that references this viewpoint aims to increase the likelihood of girls remaining engaged with computing (Porter et al., 2013), although it does not address the issue that there are too few girls entering the pipeline at the outset. A separate metaphor suggests that improving girls' participation in computing is like "unlocking the clubhouse" to create more inclusive spaces (Fisher & Margolis, 2002). Through their research and work at Carnegie Mellon University, Margolis and Fisher increased the proportion of female computing students from 7% in 1995 to 42% in 2000 through a multi-faceted set of interventions that included contextualising learning and broadening computer science stereotypes. However, the clubhouse metaphor is not widely understood outside of the context of the USA, which limits the generalisability of the findings.

2.2 Pair programming

Pair programming has been recognised as an important collaborative approach in industry and education for the past two decades (Hanks et al., 2011). The approach has been used in both agile software development and education to improve code quality (McDowell et al., 2006). Similar findings in K-12 (primary and secondary) environments demonstrated that pair programming generally increased pupil attitudes and confidence toward computing (Denner et al., 2014). Pair programming activities in school have a defined structure in which two pupils work together at a single computer to jointly create a computer program (Werner & Denning, 2009). One pupil takes on the role of 'driver', has control of the keyboard and mouse, and writes the code (Denner et al., 2014). The second pupil is the 'navigator', who reads out any instructions, monitors the code for errors, and points these out to the 'driver' (McDowell et al., 2006). Pupils regularly swap roles after a designated period of time, so that they both perform both roles equally. The teacher's role includes training the pupils in successful pair interactions and ensuring that pairs rotate regularly and fairly. The success of pair interactions is actively managed by the teacher as well as being evaluated by the pairs themselves (Williams et al., 2008).

Research that builds initial studies into pair programming is largely concerned with either improving the *process* of pair programming, or the analysing the *behaviours* that pupils exhibit when collaborating. Studies that explore the *process* of pair programming seek to better understand instructional strategies that can be used. For example, students in 4th and 5th grades were found to prefer two-computer pair programming because they felt more independent (Tsan et al., 2020) and a study with 6th grade students found that semi-free switching between roles in pairs led to increased achievement in a post-test of computing concepts (Zhong et al., 2017). However, the focus on student outcomes in these studies means that the role of the teacher in setting up effective pair programming activities has not yet been fully explored. Although the pair programming approach is a specific pedagogy employed in computing, primary (K-5) teachers use paired work in other subjects too and less is known about how teachers can use their pedagogical content knowledge (PCK) (Carlson et al., 2019) to assign effective pairings.

Recent work suggests that both teachers and students perceive pair programming as an equitable method for learning to write programs (Graßl & Foster, 2024), although when pair programming interactions have been investigated, some disparities emerge. Four different types of behaviour have been observed between pairs in an after-school setting with pupils aged 11 - 14: game interaction, when the pair discussed designing or programming the game; non-game interaction, when the pair discussed or reacted to non-task activities; third person present, when the pair interacted with a teacher or peer; and no interaction, when the pair did not interact (Campe et al., 2020). The most common interaction was about the game, although this made up less than half of all the interactions. Pair programming has also been found to lead to inequitable relationships between 11 - 12-year-old students if pairs decide to focus on completing programming tasks quickly (Lewis & Shah, 2015). These variations in pair interactions suggests that pair programming requires careful training of teachers to ensure that all elements of the pedagogy are understood and applied, without bias.

2.3 Pair programming and gender

An emerging body of evidence suggests that collaborative teaching approaches can engage more girls with computing (Tsan et al., 2016). This is of particular interest when learning to write computer programs, which can be seen as the most difficult aspect of the computing curriculum for learners (Kallia & Sentance, 2018). Introducing a shared, group approach requires a shift from traditional computing pedagogy. Learning to code changes from a series of tasks undertaken by individuals, to a sociocultural experience in which pupils work together to create and share digital content (Kafai & Burke, 2013). Talk and discussion promote a social construction of knowledge; according to sociocultural theory, a child's development involves social interaction, dialogue, and mediated activity between learners and with their teachers (Vygotsky, 1978). Gender differences have been observed where girls are more likely than boys to be motivated by social dimensions of learning (Korpershoek et al., 2021) and to express a preference for working toward social goals (Hijzen et al., 2006). Research focusing on girls' engagement has shown that pair programming particularly impacted K-12 girls' enjoyment (Liebenberg et al., 2012) and interest (Werner et al., 2004) in programming. This suggests that pair programming has the potential to be used as an inclusive pedagogy to benefit girls’

perceptions of computing, whilst also supporting all learners. Female undergraduate students also report positive outcomes from using a pair programming approach on their confidence levels, although the literature attributes this to two different mechanisms: either through social engagement and peer learning (Ying et al., 2019) or through girls observing a similarity in knowledge with peers (Yates & Plagnol, 2022). Less is known, however, about K-5 teachers' and pupils' perspectives on the mechanisms that cause these outcomes.

The research described above indicates that pair programming may be an effective way to improve the motivation of girls to continue to study computing at school. However, many of these studies are small in scale. In this study, the first intention was to consider the impact of pair programming in a larger-scale study using a rigorous quantitative method, and then to explore a smaller group of teachers' and pupils' views about which aspects of pair programming are most beneficial for girls to contribute towards better understanding the mechanisms that increase girls' confidence through pair programming activities.

The study described in this paper has the following research questions:

RQ1: How does introducing pair programming in Grades 3-5 impact on girls' attitudes towards computing?

RQ2: What are the experiences of teachers and pupils in computing lessons when using a pair programming approach?

RQ3: In what ways do teachers and female pupils think that pair programming activities increase girls' confidence in computing?

3. The Study

To address the research questions, a cross-organisational research group designed a mixed-methods study to explore the effect of teaching computing using pair programming on primary school pupils' attitudes toward computing. The study had three components:

- A small-scale preliminary study, akin to a pilot, to trial ways of introducing teachers to pair programming and evaluate the impact on a small scale.
- A large-scale main study, designed as a randomised controlled trial (RCT).
- A qualitative evaluation of the implementation fidelity and experience of the intervention with a small sample of schools.

The study was externally funded and formed part of a larger programme of separately delivered interventions. The three parts of the study were conducted by a multi-organisation research team, of which the authors are a part, between 2019 and 2022. The authors are part of the organisation that conducted the preliminary study, recruited schools, designed the intervention and provided support to participating schools for the large-scale main study. Data were collected and analysed by a second organisation. The two organisations worked closely together, and the authors recruited and liaised with participants throughout the duration of the study; in this way, all partners were familiar with and consulted about the processes used for data collection and analysis. In this paper, we have drawn on interpretations made by the second organisation, and added our own where appropriate, to situate the findings within the pair programming literature.

3.1 Preliminary study

A preliminary study, prior to the RCT, was conducted in 2019 with eleven primary schools to identify teachers' perspectives and experiences of using pair programming and incorporate these into the training materials for the main study (Leonard et al., 2021). One or two teachers from each school attended a one-day, in-person training day. The training was designed and delivered by members of the wider research team who had previous experience teaching in primary and secondary schools, introducing teachers to the pair programming methodology chosen for this study. Teachers then worked in pairs and practised the approach with a sample activity using the visual programming language Scratch[1]. Scratch was chosen because it is commonly used in primary schools in England.

[1] https://scratch.mit.edu/

With some attrition, the final sample included 10 primary schools, comprising 2 independent and 8 state-funded institutions, all of which were mixed-sex. Among them, 9 schools were situated in urban locations. These schools taught their usual programming lessons using the pair programming approach to 356 Year 6 pupils (171 female and 185 male) between January and March 2020.

The research design included an interview with participating teachers in March 2020 once they had used the pair programming approach for a minimum of six weeks in lessons. However, due to the onset of the Covid-19 pandemic, a period of emergency school closures meant that only one teacher was interviewed.

3.2 Main study: randomised controlled trial

A randomised controlled trial (RCT) is a rigorous tool to examine causal relationships between an intervention and outcome. The act of randomisation balances participant characteristics (both observed and unobserved) between the groups, allowing attribution of any differences in outcome to the study intervention (Hariton & Locascio, 2018). In this case, the RCT was designed to evaluate the impact of a 12-week intervention in school. It was funded by England's ministry of education, positioned as one pilot study amongst other studies to establish whether there was any evidence for a single intervention that might reduce gender imbalance in computing. The intervention is described in Section 4.1, and was developed by the authors and other colleagues, with the RCT itself being designed and implemented by another organisation within the broader team. The outcome measures for the RCT were: a) pupil scores on the Student Computer Science Attitudes Survey (SCSAS) (Haynie & Packman, 2017) which captures pupils' attitudes towards computing and b) pupils' response to a single item survey measure of whether the pupil plans to continue study computing.

3.3 Qualitative evaluation study

The qualitative study was, in essence, a process evaluation (Humphrey et al., 2016), which was conducted with a small number of schools to check for implementation fidelity and schools' experience of the intervention. It examined the mechanisms of change and the diversity of

implementation and intervention delivery. A case study approach (Merriam, 1998) was used which aimed to capture the range and diversity of participant experiences. Individual, in-depth semi-structured interviews lasting between 30 – 45 minutes were conducted with one teacher from each of the case study schools (see Table 1), to explore their experiences of the intervention and any factors that influenced their ability to implement the intervention with their pupils. Group discussions were conducted with pupil focus groups at two case study schools, each lasting about 20 minutes. Pupils ranked different skills by importance for computing, discussed whether statements about computing (e.g., "boys and girls are equally likely to have computing as their favourite subject") were true or false, and completed sentence starters related to pair programming and computing lessons more generally. Additionally, lesson observations were conducted in the same two schools just before the focus groups took place, so that the pair programming lessons could be referenced in the discussions.

3.4 Participants

To assist recruitment to the RCT, a third-party paid-for marketing campaign was used to maximise representative coverage across schools in England. All primary schools in England were eligible, as long as they had female pupils from either Year 4 (aged 8 - 9 years old) or Year 6 (aged 10 - 11 years old). All schools that entered the sample did so voluntarily.

The 116 participating schools were randomly divided into a control group of 58 schools, who taught computing to a `business as usual' model, and a treatment group of 58 schools, who delivered an intervention. Researchers conducting the evaluation used school reference numbers as unique identifiers to assign schools randomly to either group. Following randomisation, balance checks on other school-level variables were carried out using the school's performance status and a standard proxy for measuring socioeconomic status. Pupils were blind to allocation during the programme and during outcome data collection. Teachers were not blind to allocation; they were responsible for delivering the materials, and were aware that there was both a control and a treatment group and of which group their school was in. Data was collected for both boys and girls, but only data from girls was analysed for primary and secondary analyses in the main study.

Ethical processes were adhered to for all three parts of the study. Schools shared information sheets and withdrawal forms with all parents of pupils in participating classes. These letters explained the purpose of the research and what taking part would involve for parent (where relevant) and their child. Parents were invited to withdraw their child from the RCT if they did not want them to take part. Pupils whose parents had withdrawn them from the RCT were still able to engage in the computing activities themselves, but did not take part in any of the evaluation activities and no data on these pupils was collected. All participants in the interviews and focus groups were informed that their participation was entirely voluntary and that they could withdraw at any point.

For the qualitative evaluation, the authors recruited four state-funded schools from the treatment group using sampling criteria including geographical location and a proxy indicator of pupil socioeconomic status to ensure a mix. Quality assurance mechanisms indicated schools had good or excellent provision. Participant details are shown in Table 1.

Table 1. Sample of Schools, Teachers and Pupils Participating in the Qualitative Study

School	Teacher	Pupils
S01	Computing specialist teacher More than 10 years' teaching experience Male	6 year 4 pupils (8 - 9 years old) 4 female (range of confidence in computing) 2 male (range of confidence in computing)
S02	Computing specialist teacher More than 10 years' teaching experience Female	5 year 4 pupils (8 - 9 years old) 3 female (range of confidence in computing) 2 male (range of confidence in computing)
S03	Classroom teacher with curriculum responsibilities for computing Fewer than 10 years' teaching experience Female	None

S04	Classroom teacher with curriculum responsibilities for computing Fewer than 10 years' teaching experience Male	None

3.5 Data analysis

As described above, the RCT used the SCSAS survey tool, which is a validated survey tool designed to measure pupil attitudes towards computing (Haynie & Packman, 2017). It has a high level of within-construct consistency with the alpha value of the five sub scales ranging from 0.85 to 0.93. The language of the SCSAS questions was adapted to a) ensure the questions used language that pupils in schools in England would be familiar with and b) ensure that pupils at primary levels would be able to independently understand what the questions were asking them (e.g., by replacing the word 'peers' with 'friends').

Data cleaning of survey data ensured any data points deemed potentially unreliable were removed; for example, all data was deleted for pupils who had answered in a straight pattern (e.g., a survey with the answer 'Strongly disagree' for every question of the SCSAS). The final data set consisted of (1) data from female pupils who had completed the endline survey matched to their baseline data and (2) data from female pupils who had completed only the endline survey, using a multistep matching process to match as many baseline and endline surveys as possible (Kelly et al., 2022).

The model analysing the SCSAS scores used a linear OLS (Ordinary Least Squares) regression, with the model which had as outcomes the intention to study computing using logistic regression. All analyses were conducted on an Intention to Treat (ITT) basis, meaning that outcomes were analysed on the basis of the groups that teachers and pupils were randomly allocated to, regardless of their compliance with the intervention. All models included the following covariates: baseline SCSAS score, school performance rating, and a proxy value for socioeconomic status. Their inclusion

increases the precision of the impact estimates. All planned covariates were checked for missing data pre-analysis. Given that the endline data would likely include some pupils who were not included in the baseline dataset, pre-trial decision rules were specified for dealing with missing data as baseline scores on the SCSAS were to be used as a covariate in the analysis. More details on the data analysis are available in a technical report (Kelly et al., 2022).

For the qualitative data analysis, the data were transcribed where necessary and analysed using the Framework Approach (Ritchie et al., 2003). This involved summarising transcripts and notes into a matrix organised by themes and sub-themes (columns) as well as by individual cases (rows). This is particularly useful when multiple researchers are working on data (Gale et al., 2013). Lesson observation data was used to triangulate themes arising from interview data that related to pupil engagement. The second organisation conducted case and theme analyses with a focus on providing rich descriptions of participating experiences, whilst looking for explanations and linkages within and across participant groups, and these were then synthesised against the literature by the paper authors.

3.6 Reliability and Validity

Efforts were made to ensure the RCT was sufficiently powered to detect an effect size reliably. For each of the trials, power calculations were conducted based on informed assumptions about: the number of girls per-school per-year and the proportion of girls studying CS as an elective (source: Ministry of Education in England); the intraclass correlation coefficient (ICC) (source: comparable clustered RCTs previously carried out by the team conducting the RCT evaluations); the explanatory power of the variables included in the model at baseline (source: baseline data). The target number of schools to recruit included with a 40% attrition buffer to detect an estimated minimum detectable effect size of .05 along the 1-4 SCSAS scale, and a 10 percentage point increase in intention to study computing. However, due to higher than expected levels of attrition amongst the control group schools, the final analytical sample was not sufficient to detect an effect at the originally targeted effect size.

The Framework Approach (Ritchie et al., 2003) used for the qualitative data analysis ensures reliability through collaborative consensus, rather than using inter-rater reliability (Gale et al., 2013). Team discussions, transparent documentation and reflexivity were used to iteratively refine and agree codes, which allows for a nuanced understanding of the data while maintaining rigor.

All schools that entered the samples for both the main study and the qualitative research did so voluntarily, which has implications for the external validity of the findings. Schools that volunteer to participate in research are likely to be more enthusiastic about the intervention than an average school, and this may interact with the treatment effect to compound any effects.

4. The pair programming intervention

4.1 Intervention design

For the RCT intervention, a full set of teaching materials, including lesson plans, slide decks, example Scratch projects and seating plan templates were written by learning experience designers who had previously taught computing in schools. A panel of current primary educators reviewed the teaching materials for age-appropriateness and accuracy, and following this review some minor revisions were made.

The materials comprised twelve 1-hour lessons, which were divided into two units of work, each containing six lessons. Unit 1 started with an introductory lesson where pupils learned about pair programming, practised working in pairs, and were introduced to the pair programming map. This was followed by five lessons focussing on the creation of drawings in Scratch. In Unit 2, pupils took part in six lessons learning how to create simple animations in Scratch. In every lesson, there was a check-in for pupils to reflect on the effectiveness of their paired working. After each check-in, each pair of pupils were prompted to build up a set of rules for successful pair programming work. An overview of the two units is shown in Tables 2 and 3.

Table 2. Drawing Shapes (Unit 1)

No	Lesson	Learning objective	Example activity
1	Introduction to pair programming	To explain how working in pairs can help you learn to program	Pupils work in their allocated pairs to create a name for their pair and identify some ways that they plan to work together successfully.
2	Shapes	To explain how basic shapes can be drawn using a series of movements in a computer program	Pupils work in pairs to predict what shape a Scratch program will make and then modify it to create their own shape.
3	Pen marks	To explain how an input triggers an output	Pupils work in pairs to use an Event block and the Pen blocks to draw a shape. Pairs also think how they helped each other and how they might help each other more in the next task.
4	Repeat	To explain how to create shapes and patterns through repetition using the 'repeat' Control block	Pupils work in pairs to use the Repeat block to draw a square.
5	Tidy up time	To explain how to organise a program using subroutines from My Blocks	Pupils work in pairs to create a procedure to draw a shape using the My Blocks feature. They also discuss how both partners can contribute equally to the programming tasks.

No	Lesson	Learning objective	Example activity
6	Geometric line art	To explain how to create patterns through repetition using the 'repeat' Control block	Pupils work in pairs to use the skills they have learned in this unit to create geometric art shapes.

4.2 Implementation of intervention

The pair programming approach used in the main study was 'driver-navigator' (Zhong et al., 2017) with two pupils sharing a computer. Teachers were asked to make sure pupils switched roles every five minutes and guided to use a digital timer to ensure this was done accurately.

Pairs were decided on in advance by the teacher, using guidance such as pairing pupils of similar skill levels or pupils who had similar attitudes towards classwork. Teachers were advised to avoid swapping pairs and instead to plan bespoke activities to improve pair work if they noticed that pairs were working ineffectively.

Table 3. Programming Animation (Unit 2)

No	Lesson	Learning objective	Example activity
7	Two sprites	To explain how to create animation using two or more sprites and the 'move' and 'turn' Motion blocks	Pupils work in pairs to create a Scratch program that animates two sprites to dance together.
8	Park life	Pupils create simple interactive animations and explore selection using the 'if () then' Control block	Pupils investigate a pre-made Scratch program which uses selection to determine which animation effect is run.

No	Lesson	Learning objective	Example activity
9	Sea life	To explain how to create animations using the 'if () then, else' Control block	Pupils work in pairs to animate an underwater scene. They also discuss the 'Navigator' role and identify ways that they can improve giving instructions.
10	Artist's life	To explain how to create a detailed graphic animation using costumes	Pupils draw inspiration from real-life 'flip books' to add costumes to their animations. They also generate a list of six reasons to use pair programming.
11	Give life to	To explain how to prepare to create an animation using sequence in a storyboard	Pupils use a pre-made storyboard template to plan an animation of their choice. They also review another pair's map of pair programming rules.
12	Let's create	To explain how an interactive animation incorporates the three programming constructs — sequence, repetition, and selection	Pupils celebrate their successes in pair programming. They also program a Scratch animation based on their storyboard from lesson 11.

Each pair was given a printed document called 'Map to successful pair programming' to complete during the twelve lessons (see Figure 1 for an example). Rules were set out for the 'driver' role, the 'navigator' role and for the pair overall.

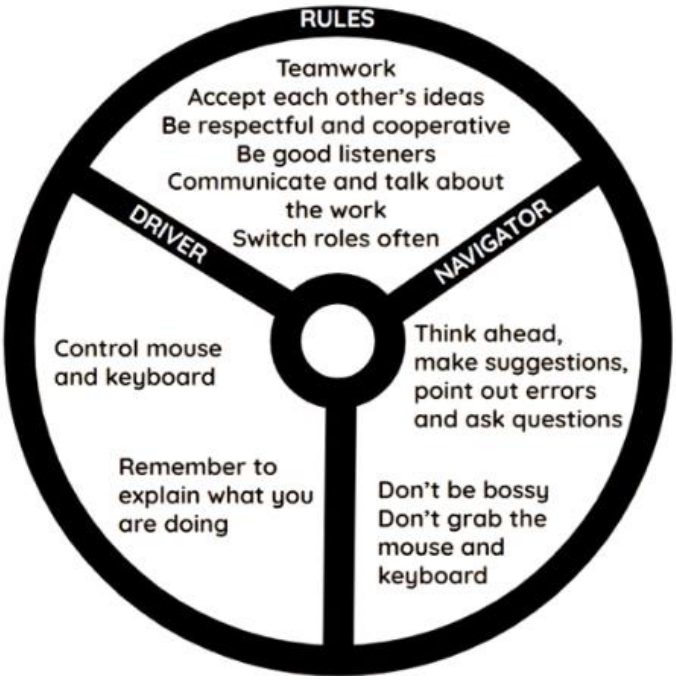

Figure 1. An example of a completed steering wheel with guidance for successful pair programming work

4.3 Teacher Preparation

The planned face-to-face training for teachers participating in the RCT was moved to an online format due to Covid-19 restrictions affecting travel and the mixing of households. All teaching staff from schools in the treatment group were required to take a mandatory, 2-hour online training course, and schools could claim back any incurred costs. Professional development and reimbursement were available to all teachers and teaching assistants involved in teaching the lessons to ensure consistent use of the pair programming approach.

The training course was divided into two parts. In part 1, the study was introduced alongside existing research into the effectiveness of the pair programming approach. In part 2, teachers were introduced to the units of work and prompted to explore the teaching materials. A range of resources were used, including a video of pair programming in the primary classroom from the preliminary study. Teachers were then given clear instructions on how to administer the evaluation surveys at the beginning and the end of the 12 week unit.

5. Results

In this section, the findings from the intervention are shared. The first research question sought to explore how girls' attitudes to computing were impacted by introducing pair programming and is answered by the RCT results presented in section 5.1. We address research question two about teacher and pupil experiences in computing lessons using a pair programming approach through qualitative findings presented in section 5.2. Finally, in section 5.3 we present data relating to our third research question about teachers' and pupils' perspectives concerning the impact of pair programming activities on girls' confidence.

5.1 RCT results

From the 116 schools, the analysis of the RCT used 990 responses of pupil data of attitudes towards computing, and 994 responses of pupil data of intent to study computing. These are reported separately in section 5.1.1. and section 5.1.2

5.1.1 Effect on attitudes towards computing

Results from the SCSAS survey comparing data from pupils in the control and treatment groups are shown in Table 4. A pre-specified multiple imputation model to account for missing baseline or endline data (Kelly et al., 2022) found a difference of 0.046 points (p=0.331) on a 1-4 scale for the intervention, which was not statistically significant at conventional significance levels. While the difference was positive, it was small in magnitude relative to the 1-4 scale and might not represent a meaningful shift in girls' attitudes even if it were significant. A standard error of 0.047 was calculated which indicates a high precision in the sample mean estimate and suggests that the data closely approximates normality.

Therefore, the intervention's impact on girls' attitudes toward computing, as measured by the SCSAS, was not statistically significant compared to the control group.

Table 4. Study Results for Girls' Attitudes Towards Computing

Measure	Control group mean	Treatment group mean	ETE (SE)	*p*
SCSAS score	2.80	2.92	0.046 (0.047)	0.331

***Note. N* = 990. SCSAS = Student computer science attitude survey. ETE = estimated treatment effect. SE = standard error.**

5.1.2 Effect on stated intention to study computing in the future

Results for the stated intention to study computing in the future are shown in Table 5. The pre-specified multiple imputation model to account for missing baseline or endline data (Kelly et al., 2022) found a 4.3 percentage points (p=0.453) difference in favour of the intervention compared to the control group, which, although positive, is not statistically significant by conventional standards, and cannot be confidently attributed to the intervention rather than random chance. Therefore, the RCT revealed no conclusive evidence that the intervention positively affected girls' intentions to pursue computing studies in the future, as compared to the control group.

Table 5. Study Results for Girls' Self-Reported Intent to Study Computing in the Future

Measure	Control group mean	Treatment group mean	ETE	*p*
Positive self-reported intent to study computing in the future	45%	49.8%	4.3pp	0.453

***Note. N* = 994. SCSAS = Student computer science attitude survey. ETE = estimated treatment effect.**

5.2 Themes from qualitative evaluation study

For the qualitative evaluation, the framework analysis generated four overarching themes relating to the implementation of the intervention, encompassing: fidelity of the paired work, feasibility of the paired work, teacher and pupil experiences of paired working, and mechanisms activated by the intervention.

Fidelity of paired work: Schools in the treatment group largely implemented the pair programming approach as it had been intended. Interviewed teachers generally set up the pairs so that they were mixed-gender, although that was not always possible given the gender balance of the class. All four teachers explained that they spent time carefully pairing pupils according to a variety of different criteria, including matching pairs by computing attainment, creating mixed attaining pairs or using the same pairs as in maths lessons.

"I thought very carefully about pairing the children up from my knowledge of, how they are, their current abilities, where they progressed to in terms of computing." (Teacher: S01)

Within the four case study schools, fidelity to having a driver and navigator with clearly defined roles was also high: pupils understood the different responsibilities that went with the different roles and responded promptly to the indication that it was time to swap roles (normally by changing seats).

"Now they've got used to the system, this is the lesson and this is what we're doing. That's been good." (Teacher: S02)

Feasibility of paired work: Characteristics of the learning environment such as pupil behaviour and pupil familiarity with routines affected the ease with which teachers could implement the intervention. Observations that took place towards the end of the twelve-week unit showed that pupils knew where they should be sitting, who their partners were, which partner was starting in which role, where to find the Scratch project and how to switch places. It is likely that earlier in the unit, teachers spent more time establishing routines and expectations.

"There were a few times...where they weren't collaborating properly...At the beginning [of the unit] we had to have at least two or three [lessons]....just reminding them what the role of the navigator and what the driver was." (Teacher: S04)

Overall, comments from the teachers implied that pair programming was well suited to computing because the clearly defined roles provided a strong collaborative environment that supported learning.

"I don't know why I've never thought to do computing like that, actually because it's a really good vehicle for the fact that there are two roles, clearly defined. There's all your conversation and knowledge comes through that, and then they're both equally having a turn." (Teacher: S04)

Teacher and pupil experiences of paired work: Data collected from discussions with pupils and teachers, and observations of lessons, showed that pupils enjoyed working with a partner. For example, a teacher in school S04 explained that *"they [the pupils] really liked working together as driver and navigator. They liked the aspect of swapping around."*

Observations of the paired work showed that the discussions between pairs often focussed on task planning, including naming the blocks of code to use in Scratch, for example:

Boy: "What do you want him to say? Shall we try..let's set off?"

Girl: "Yeh but we need a start block first" (Male and female pupils: S02)

Pupils' comments suggested that using the pair programming approach had been enjoyable, although they also critically evaluated the circumstances that might mean paired work was less effective.

"I like working with both [both in a partner and by yourself] because when you do pair programming you're collaborating with your partner, making links and you have to tell them what to do. But if you have a really good idea and then they put the wrong thing in the wrong place, it's quite annoying." (Female pupil: S01)

"Sometimes when you're in your pairs...it's trickier in different pairs because you don't pick your pairs - they get picked for you. Sometimes it might be easier working in one pair than another." (Female pupil: S02)

All four interviewed teachers stated that following their experiences using the pair programming approach, they intended to use it in the future for other computing lessons. For example,

"Even those who are maybe a little bit more reluctant...those who put their hands up today and said they still prefer to work independently, they are still all engaging quite clearly in that with their pair and doing it really, really well. However much they say they prefer working independently, I think they clearly showed how much they enjoy it, engage with it. And you know they're achieving with it - so we should be doing this." (Teacher: S01)

"I felt like some of my girls were really quite [makes bored noise] at the beginning, and by the end of it, especially when we did the second unit about the animation, loved it. They absolutely loved it." (Teacher: S03)

5.3 Confidence

We aimed to better understand the mechanisms through which pair programming activities would increase girls' confidence in programming and so synthesised the data collected from the qualitative evaluation study by the independent evaluators to gather evidence on student and teacher views.

Both interviewed teachers and pupils felt that having the support of a partner boosted girls' confidence because they could work together to overcome challenges.

"I do think that having that equal time to have a go at both, thinking of the girls I've got, will have helped my girls, because they lack a bit of confidence. They were learning very quickly that actually 'Yes, we are sure. We can do this.' " (S03)

Girls said that they liked having a partner because *"someone else … could help me if I needed help"* (S02: female pupil) and that *"if you're stuck your partner can be helpful "* (S01: female pupil).

We had hypothesised that by feeling more confident, girls would engage more in discussions with their partner and increase their subject knowledge. One interviewed teacher made this connection explicitly, explaining that the confidence that girls had through working with a partner enabled them to learn more than if the approach had not been used.

"The pair programming definitely helps. I think it boosted their confidence. They [the girls] had a partner to work with so immediately that makes it more interesting for them. I don't know, I feel like they just acquired more knowledge." (S03: Teacher)

Girls tended to be positive about the collaborative elements of the Pair Programming lessons because of the opportunity to build relationships with peers.

"If someone was your friend you'd make them be more of your friend because you'd be talking with them more, sharing their interest and knowing what they like." (S02: female pupil)

Interviewed teachers had different opinions about whether the intervention led to the same outcomes for both boys and girls. One teacher described how the engagement had increased indiscriminately of gender:

"Nothing stands out in particular. I'm not going to try and kind of conjure something. I'm pleased to say that there's been equal engagement and an equal impact on both male and female." (S01: Teacher)

On the other hand, another teacher did feel that her female pupils had started with lower confidence and so the intervention had particularly appealed to girls.

"I think the girls would have come out better from it, because of their confidence towards the subject. The boys, they liked it, but I feel like the girls were more engaged with it. I don't know if I would have seen that level of engagement from the girls, if it wasn't taught that way, because I do think a lot of mine go in on themselves when they don't know." (S03: Teacher)

6. Discussion

6.1 Explaining the RCT findings

The results of the RCT (RQ1) show that there are no statistically significant results for either of the two outcomes. These results can be explained in a number of ways. Firstly, it may be surmised that the results may not be reliable due to either the impact of the coronavirus pandemic or the design of the study. For example, one interpretation of the results could be that the COVID-19 pandemic, which had an impact on school recruitment and retention, as well as on implementation fidelity, has affected the results; this would imply that conducting the trials again might produce significant results for some of the trials. However, power calculations were used and the trials only went ahead once the number of schools needed had been recruited. In addition, some expected attrition was built into the modelling. Thus, the size of the trial should still be adequate.

Another interpretation is that the endline data might have been collected too soon to show any impact. Collecting data from students a few months later might have given more opportunity for a significant change in attitude and behaviour. Change may still occur for those pupils (and indeed their teachers) involved in the set of interventions. As often in research projects, pressure to conclude the research dictated the collection of only one set of endline data at the end of each intervention: collecting attitudinal and behavioural change later on might have given more opportunity for significant change to be visible.

Without any evidence that the design or implementation of the study was faulty, the study's findings do indicate that including a pair programming methodology alone for 12-weeks does not make a

significant difference to girls' attitudes towards computing. It may be that making more changes to pedagogical approaches, as well as for a longer period of time, might be needed to make any noticeable impact. However, the study as implemented has not been able to show any positively significant (or positively negative) results.

6.2 The use of RCTs in education

In some fields, including education policy, the RCT is seen as the `gold standard' of evaluation as randomisation eliminates much of the bias inherent with other study designs (Xiao et al., 2020). The RCT approach, drawn from a positivist view of education, is not universally popular in education, as others arguing that while we need to provide reliable evidence for research findings, the RCT should not be elevated above any other methodology (Morrison, 2020).

In classrooms and schools, there are multiple variables that are difficult to control for, and a more interpretivist lens is commonly used to interpret findings (Torgerson & Torgerson, 2001). In the study described in this paper, models of analysis were carefully developed to control for multiple factors such as whether the school was in a low-income area, the overall performance of the school as judged by an external body, and the baseline scores of the pupils. There will always be other variables that cannot be controlled for; for this reason it is useful to consider the qualitative research as triangulation for the quantitative findings. Indeed even positive RCT results need to be regarded carefully in education:

"*Just because research has shown that such-and-such might `work' in a such-and-such research setting, be it contrived or naturalistic, this is no reason to believe that it will work in a different temporal, locational, contextual setting, or even the same setting, a second time."* (Morrison, 2020, p. 11).

6.3 Reporting non-significant RCT results

Researchers hesitate to report inconclusive findings in academic publications (Coldwell & Moore, 2024). However, the authors consider that reporting non-significant findings will support further research in this area. Not reporting inconclusive or negative findings can lead to publication bias (Haden, 2019, Coldwell & Moore, 2024). Selective reporting of scientific findings is sometimes

known as the file drawer problem (Rosenthal, 1979), describing the tendency of researchers to publish positive results much more readily than negative results, which `end up in the researcher's drawer'. This leads to publication bias, where only positive results are reported in the literature. In this case, it is important to be able to suggest some interpretations of the quantitative results, and explanations for some discord with the (albeit low volume) qualitative results, which will be discussed next.

6.4 Aspects of pair programming emerging from the qualitative evaluation

Despite the inconclusive results of the RCT on girls' attitudes towards and intent to study computing, the qualitative data did identify some mechanisms through which the intervention might have led to the intended effect on girls' attitudes towards computing. This inconsistency may be due to the reliance on pupil-reported quantitative data immediately after the intervention had taken place rather than observable, independent indicators of effect of the RCT. Furthermore, the high stated intention to study computing scores suggest that the sample of participants could have already have had relatively high engagement with computing, thus making it more difficult for the RCT to detect an impact. The qualitative data, although small in scale, gives us another set of data with which to investigate aspects of pair programming and highlight in more detail which mechanisms may have been most effective. In the case studies, teachers were enthusiastic about using the pair programming approach and provided insights which have implications for teacher professional development and future implementations of pair programming with K-5 learners. Thus the qualitative research indicates that introducing pair programming can effect some changes and reflections for teachers and pupils. These can be described under three headings:

1. Development of Pedagogical Content Knowledge (implementation)
2. Confidence and self-efficacy (teacher and pupil experiences)
3. Collaboration and teamwork (teacher and pupil experiences)

6.4.1 Pedagogical content knowledge

RQ2 sought to better understand how teachers pair pupils together for pair programming activities, In the same way as other studies (e.g. Tsan et al., 2020; Vandenberg et al., 2023), the instructional design of our resources included the freedom to pair pupils by drawing on teachers' prior knowledge and insights into their pupils' behaviours and teachers' existing pedagogical knowledge of classroom management (Magnusson et al., 2002). This resulted in a variety of criteria for pairing (mixed attainment pairs, mixed gender pairs and reusing pairs from maths lessons) but a high level of fidelity to the pair programming approach across all four schools.

When viewed through the lens of the Refined Consensus Model of Pedagogical Content Knowledge (PCK) (Carlson et al., 2019), teachers used pedagogical reasoning that drew on their *enacted PCK* to pair pupils based on their existing knowledge of what works in other subjects (e.g. mathematics), or their *personal PCK* to create mixed gender pairs based on their beliefs of improving the gender balance in computing. Similarly, teachers also used their *enacted PCK* of embedding classroom routines to help lessons run smoothly. Professional development in the pair programming approach can mobilise teachers' existing *enacted PCK* of successful collaboration strategies and classroom routines to increase the potential for effective implementation.

6.4.2 Confidence and self-efficacy

Findings from the qualitative study showed having a partner's support bolstered girls' confidence, a finding that is consistent with prior research (e.g. Werner et al., 2004). Positive emotions such as confidence towards a subject, coupled with a belief in one's ability to succeed in related tasks, signify self-efficacy (Bandura, 1977). In computing in general, there are gender differences evident when pupils self-assess their abilities, with girls underestimating their performance compared with boys, who demonstrate more accurate self-assessments (Kallia & Sentance, 2018). Our results showed that girls benefited from working with a partner to boost their confidence through social engagement and knowledge-building, rather than comparing themselves with peers. This may be because of the age of

the pupils involved in the study, and stereotypes about boys' superior ability in computing have not yet developed. Interviewed teachers and pupils provided evidence to support findings from prior work, that the girls felt more confident engaging with computing and took part in discussions during paired work which led to increased subject knowledge (Ying et al., 2019).

Pupil and teacher comments from our findings suggest that using the pair programming approach may enhance both boys' and girls' self-efficacy. This emergent finding is noteworthy as a strong sense of self-efficacy in computing is connected with pupils' decisions to pursue further studies in the field in both K-12 and undergraduate education (Mishkin, 2019; Aivaloglou & Hermans, 2019). It is crucial that computing teachers are trained in and can use approaches that aim to benefit girls' engagement in computing education and careers to ensure an equitable approach for all pupils, and that these approaches begin in K-5 education, so that all pupils have positive initial experiences of computing.

6.4.3 Collaboration and teamwork

The qualitative findings also showed that girls valued collaboration and teamwork, an additional mechanism for engaging girls in computing lessons that we had not planned to investigate, but which was reported by teachers and pupils during interviews and focus groups. This result is in keeping with general education research indicating females from some Western cultures may be more motivated by a social rather than a competitively oriented learning context (Korpershoek et al., 2021; Hijzen et al., 2006), and builds on sociocultural theories proposing that pupils use social interactions to learn with and from their peers and educators (Vygotsky, 1978). Similar indications of females preferring to learn in a more social and collaborative setting in CS have been inferred from instructional approach research into pair programming (Denner et al., 2014; Liebenberg et al., 2012). This reinforces calls in the field to take a more sociocultural approach in CS (e.g. Guzdial & Tew, 2006; Kafai & Burke, 2013; Faraon et al., 2020; Vrieler & Salminen-Karlsson, 2022).

6.5 Limitations

This study has potential limitations, including the lack of gender-specific analysis to compare boys' and girls' attitudes. This introduces the potential risk of causing a 'backfire' effect, where well-

intentioned interventions have adverse outcomes for certain students. Future research should address these by examining gender differences and considering diverse educational contexts. By addressing these areas, future work can develop more inclusive and effective pair programming strategies that promote collaborative learning and student success.

7. Conclusion

In summary, this paper describes a study involving 116 primary schools and 990 female pupils which investigated the impact of a 12-week pair programming intervention on pupil's attitudes to computing and their intention to study it further. The main study (RCT) did not find any statistically significant results in terms of the two outcome measures. However the qualitative evaluation showed that both teachers and pupils enjoyed working in pairs.

We can speculate about the null result from the RCT that while it may be that there is no causal link between using the pair programming approach and an increase in girls' attitudes towards computing, it may also need a longer period of time (greater than 12 weeks) to be evident. This might mean that the use of pair programming is implemented for longer, or that the survey and follow-up evaluation is conducted later on, forming a more longitudinal study. Another suggested explanation is that the pair programming approach needs to be combined with other strategies to achieve a positive effect. Further research could help us in establishing which of these explanations may be most likely. In any case, this research study reaffirms the fact that gender balance in computing is a deeply systemic and societal issue that requires change beyond the classroom to improve.

Although the RCT showed no statistically significant changes in attitudes or intent, qualitative data revealed that the interventions were engaging and enjoyable, with increased confidence and engagement in discussions among the girls. These findings highlight the ongoing societal and systemic barriers in computing education while indicating the positive reception of pair programming among participants.

Acknowledgements

We are grateful to the Department for Education in England for funding this research. Additionally, we would like to thank Julia Ryle-Hodges and the team at Behavioural Insights Team for their partnership on the project, Hayley Leonard, Oliver Quinlan, Emma Posey, Caitlyn Merry and Rik Cross for their contributions towards the project management and design of the intervention resources, and the teachers involved for their active engagement.

References

Aivaloglou, E., & Hermans, F. (2019). Early programming education and career orientation: The effects of gender, self-efficacy, motivation and stereotypes. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 679–685. https://doi.org/10.1145/3287324.3287358

Bandura, A. (1977). Self-Efficacy: Toward a Unifying Theory of Behavioral Change. *Psychological Review*, *84*(2), 191–215. https://doi.org/10.1037/0033-295x.84.2.191

Beyer, S., Rynes, K., Perrault, J., Hay, K., & Haller, S. (2003). Gender differences in computer science students. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, 49–53. https://doi.org/10.1145/792548.611930

Butler, D. (2000). Gender, girls, and computer technology: What's the status now? *The Clearing House*, *73*(4), 225–229. https://doi.org/10.1080/00098650009600950

Camp, T. (1997). The incredible shrinking pipeline. *COMMUNICATIONS OF THE ACM*, *40*(10). https://doi.org/10.1145/262793.262813

Campe, S., Denner, J., Green, E., & Torres, D. (2020). Pair programming in middle school: Variations in interactions and behaviors. *Computer Science Education*, *30*(1), 22–46. https://doi.org/10.1080/08993408.2019.1648119

Carlson, J., Daehler, K. R., Alonzo, A. C., Barendsen, E., Berry, A., Borowski, A., Carpendale, J., Kam Ho Chan, K., Cooper, R., Friedrichsen, P., Gess-Newsome, J., Henze-Rietveld, I., Hume, A., Kirschner, S., Liepertz, S., Loughran, J., Mavhunga, E., Neumann, K., Nilsson, P., …

Wilson, C. D. (2019). The Refined Consensus Model of Pedagogical Content Knowledge in Science Education. In A. Hume, R. Cooper, & A. Borowski (Eds.), *Repositioning Pedagogical Content Knowledge in Teachers' Knowledge for Teaching Science* (pp. 77–94). Springer Singapore. https://doi.org/10.1007/978-981-13-5898-2_2

Cheryan, S., Plaut, V. C., Davies, P. G., & Steele, C. M. (2009). Ambient belonging: How stereotypical cues impact gender participation in computer science. *Journal of Personality and Social Psychology*, *97*(6), 1045. https://doi.org/10.1037/a0016239

Coldwell, M., & Moore, N. (2024). Learning from failure: A context-informed perspective on RCTs. *British Educational Research Journal*, *50*(3), 1043–1063. https://doi.org/10.1002/berj.3960

Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education*, *46*(3), 277–296. https://doi.org/10.1080/15391523.2014.888272

Eccles, J. S. (2015). Gendered Socialization of STEM Interests in the Family. *International Journal of Gender, Science and Technology*, *7*(2), 116–132.

Faraon, M., Rönkkö, K., Wiberg, M., & Ramberg, R. (2020). Learning by coding: A sociocultural approach to teaching web development in higher education. *Education and Information Technologies*, *25*(3), 1759–1783. https://doi.org/10.1007/s10639-019-10037-x

Fisher, A., & Margolis, J. (2002). Unlocking the clubhouse: The Carnegie Mellon experience. *SIGCSE Bull.*, *34*(2), 79–83. https://doi.org/10.1145/543812.543836

Gale, N. K., Heath, G., Cameron, E., Rashid, S., & Redwood, S. (2013). Using the framework method for the analysis of qualitative data in multi-disciplinary health research. *BMC Medical Research Methodology*, *13*(1), 1–8. https://doi.org/10.1186/1471-2288-13-117

Global Education Monitoring Report Team. (2024). *Global education monitoring report 2024, gender report: technology on her terms*. UNESCO. https://doi.org/10.54676/WVCF2762

Graßl, I., & Fraser, G. Equitable Student Collaboration in Pair Programming **2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)**, Lisbon, Portugal, 2024, pp. 274-285. https://doi.org/10.1145/3639474.364008

Guzdial, M., & Tew, A. E. (2006). Imagineering inauthentic legitimate peripheral participation: An instructional design approach for motivating computing education. *ICER 2006 - Proceedings of the 2nd International Computing Education Research Workshop*, *2006*, 51–58. https://doi.org/10.1145/1151588.1151597

Haden, P. (2019). Inferential Statistics. In S. A. Fincher & A. V. E. Robins (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 133–172). Cambridge University Press. https://doi.org/10.1017/9781108654555.007

Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair programming in education: A literature review. *Comput. Sci. Educ.*, *21*(2), 135–173. https://doi.org/10.1080/08993408.2011.579808

Hariton, E., & Locascio, J. J. (2018). Randomised controlled trials—The gold standard for effectiveness research. *BJOG: An International Journal of Obstetrics and Gynaecology*, *125*(13), 1716. https://doi.org/10.1111/1471-0528.15199

Hamer, J. M. M., Kemp, P. E. J., Wong, B., & Copsey-Blake, M. (2023). Who wants to be a computer scientist? The computing aspirations of students in English secondary schools. *International Journal of Science Education*, *45*(12), 990–1007. https://doi.org/10.1080/09500693.2023.2179379

Haynie, K. C., & Packman, S. (2017). AP CS Principles Phase II: Broadening Participation in Computer Science Final Evaluation Report. In *Prepared for The College Board and the National Science Foundation*. The College Board and National Science Foundation.

Hijzen, D., Boekaerts, M., & Vedder, P. (2006). The relationship between the quality of cooperative learning, students' goal preferences, and perceptions of contextual factors in the classroom. *Scandinavian Journal of Psychology*, *47*(1), 9–21. https://doi.org/10.1111/j.1467-9450.2006.00488.x

Humphrey, N., Lendrum, A., Ashworth, E., Frearson, K., & Buck, R. (2016). Implementation and process evaluation (IPE) for interventions in education settings: An introductory handbook. *Education Endowment Foundation*.

Kafai, Y. B., & Burke, Q. (2013). The Social Turn in K-12 Programming: Moving from

Computational Thinking to Computational Participation. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 603–608. https://doi.org/10.1145/2445196.2445373

Kallia, M., & Sentance, S. (2018). Are Boys More Confident than Girls? The Role of Calibration and Students' Self-Efficacy in Programming Tasks and Computer Science. *Proceedings of the 13th Workshop in Primary and Secondary Computing Education*. https://doi.org/10.1145/3265757.3265773

Kelly, S., Ryle-Hodges, J., & Bisserbe, C. (2022). *Evaluation of Teaching Approach: Pair Programming intervention.* The Behavioural Insights Team. https://www.raspberrypi.org/app/uploads/2023/02/Gender-Balance-in-Computing-Evaluation-Report-Pair-Programming.pdf

Kemp, P. E. J., Wong, B., & Berry, M. G. (2019). Female Performance and Participation in Computer Science. *ACM Transactions on Computing Education*, *20*(1), 1–28. https://doi.org/10.1145/3366016

Korpershoek, H., King, R. B., McInerney, D. M., Nasser, R. N., Ganotice, F. A., & Watkins, D. A. (2021). Gender and cultural differences in school motivation. *Research Papers in Education*, *36*(1), 27–51. https://doi.org/10.1080/02671522.2019.1633557

Leonard, H. C., Quinlan, O., & Sentance, S. (2021, September). Female pupils' attitudes to computing in early adolescence. In *Proceedings of the 2021 Conference on United Kingdom & Ireland Computing Education Research* (pp. 1-6). https://doi.org/10.1145/3481282.3481289

Lewis, C. M., & Shah, N. (2015). How Equity and Inequity Can Emerge in Pair Programming. *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, 41–50. https://doi.org/10.1145/2787622.2787716

Liebenberg, J., Mentz, E., & Breed, B. (2012). Pair programming and secondary school girls' enjoyment of programming and the subject Information Technology (IT). *Computer Science Education*, *22*(3), 219–236. https://doi.org/10.1080/08993408.2012.713180

Magnusson, S., Krajcik, J., & Borko, H. (2002). Nature, Sources, and Development of Pedagogical Content Knowledge for Science Teaching. In J. Gess-Newsome & N. G. Lederman (Eds.),

Examining Pedagogical Content Knowledge (Vol. 6, pp. 95–132). Kluwer Academic Publishers. https://doi.org/10.1007/0-306-47217-1_4

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair Programming Improves Student Retention, Confidence, and Program Quality. *Commun. ACM*, *49*(8), 90–95. https://doi.org/10.1145/1145287.1145293

Merriam, S. B. (1998). *Qualitative Research and Case Study Applications in Education. Revised and Expanded from Case Study Research in Education*. Jossey-Bass Publishers.

Mishkin, A. (2019). Applying Self-Determination Theory towards Motivating Young Women in Computer Science. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 1025–1031. https://doi.org/10.1145/3287324.3287389

Morrison, K. (2020). *Taming randomized controlled trials in education: Exploring key claims, issues and debates*. Routledge. https://doi.org/10.4324/9781003042112

Porter, L., Bailey-Lee, C., & Simon, B. (2013). Halving Fail Rates using Peer Instruction: A Study of Four Computer Science Courses. *SIGCSE'13 : Proceedings of the 44th ACM Technical Symposium on Computer Science Education : March 6-9, 2013, Denver, Colorado, USA*, 778. https://doi.org/10.1145/2445196.2445250

Ritchie, J., Spencer, L., & O'Connor, W. (2003). Carrying out Qualitative Analysis. In J. Ritchie & J. Lewis (Eds.), *Qualitative Research Practice: A Guide for Social Science Students and Researchers*.

Rosenthal, R. (1979). The file drawer problem and tolerance for null results. *Psychological Bulletin*, *86*(3), 638. https://doi.org/10.1037/0033-2909.86.3.638

Torgerson, C. J., & Torgerson, D. J. (2001). The need for randomised controlled trials in educational research. *British Journal of Educational Studies*, *49*(3), 316–328. https://doi.org/10.1111/1467-8527.t01-1-00178

Tsan, J., Boyer, K. E., & Lynch, C. F. (2016). How early does the CS gender gap emerge? A study of collaborative problem solving in 5th grade computer science. *SIGCSE 2016 - Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 388–393. https://doi.org/10.1145/2839509.2844605

Tsan, J., Vandenberg, J., Zakaria, Z., Wiggins, J. B., Webber, A. R., Bradbury, A., Lynch, C., Wiebe, E., & Boyer, K. E. (2020). A Comparison of Two Pair Programming Configurations for Upper Elementary Students. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 346–352. https://doi.org/10.1145/3328778.3366941

United Nations Development Programme (UNDP). (n.d.). *Goal 4: Quality Education*. https://www.globalgoals.org/goals/4-quality-education/

Vandenberg, J., Lynch, C., Boyer, K. E., & Wiebe, E. (2023). "I remember how to do it": Exploring upper elementary students' collaborative regulation while pair programming using epistemic network analysis. *Computer Science Education*, *33*(3), 429–457. https://doi.org/10.1080/08993408.2022.2044672

Varma, R. (2009). Gender differences in factors influencing students towards computing. *Computer Science Education*, *19*(1), 37–49. https://doi.org/10.1080/08993400902819006

Vrieler, T., & Salminen-Karlsson, M. (2022). A Sociocultural Perspective on Computer Science Capital and Its Pedagogical Implications in Computer Science Education. *ACM Transactions on Computing Education*, *22*(4), 1–23. https://doi.org/10.1145/3487052

Vygotsky, L. S. (1978). *Mind in society: The Development of Higher Psychological Processes*. Harvard University Press.

Werner, L., & Denning, J. (2009). Pair programming in middle school: What does it look like? *Journal of Research on Technology in Education*, *42*(1), 29–49. https://doi.org/10.1080/15391523.2009.10782540

Werner, L. L., McDowell, C., & Hanks, B. (2004). Pair-Programming Helps Female Computer Science Students. *ACM Journal on Educational Resources in Computing*, *4*(1), 4. https://doi.org/10.1145/1060071.1060075

Williams, L., McCrickard, D. S., Layman, L., & Hussein, K. (2008). Eleven guidelines for implementing pair programming in the classroom. *Proceedings - Agile 2008 Conference*, 445–452. https://doi.org/10.1109/Agile.2008.12

Xiao, Z., Hauser, O. P., Kirkwood, C., Li, D. Z., Jones, B., & Higgins, S. (2020). Uncovering Individualised Treatment Effect: Evidence from Educational Trials. *IDEAS Working Paper*

Series from RePEc. ABI/INFORM Collection; Publicly Available Content Database. https://doi.org/10.31219/osf.io/8nsw4

Yates, J., & Plagnol, A. C. (2022). Female computer science students: A qualitative exploration of women's experiences studying computer science at university in the UK. *Education and Information Technologies*, *27*(3), 3079–3105. https://doi.org/10.1007/s10639-021-10743-5

Ying, K. M., Pezzullo, L. G., Ahmed, M., Crompton, K., Blanchard, J., & Boyer, K. E. (2019). In Their Own Words: Gender Differences in Student Perceptions of Pair Programming. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 1053–1059. https://doi.org/10.1145/3287324.3287380

Zhong, Baichang, Wang, Qiyun, Chen, Jie, & Li, Yi. (2017). Investigating the Period of Switching Roles in Pair Programming in a Primary School. *Educational Technology & Society*, *20*(3), 220–233. https://www.jstor.org/stable/26196132

International Journal of Computer Science Education in Schools, November 2024, Vol. 7, No. 1
ISSN 2513-8359

Students as Creators of Contexts for Learning Algorithms: How Collaborative Context Design Contributes to a Wide Range of Learning Outcomes

Jacqueline Nijenhuis-Voogt
jacqueline.nijenhuis@ru.nl
Radboud Teachers Academy
Radboud University, Nijmegen, The Netherlands

Durdane Bayram
d.bayram@tue.nl
Eindhoven School of Education
Eindhoven University of Technology, Eindhoven, The Netherlands

Paulien C. Meijer
paulien.meijer@ru.nl
Radboud Teachers Academy
Radboud University, Nijmegen, The Netherlands

Erik Barendsen
erik.barendsen@ru.nl
Institute for Science Education
Radboud University, Nijmegen, The Netherlands
Department of Computer Science
Open University, The Netherlands

DOI: 10.21585/ijcses.v7i1.190

Abstract

A context-based approach to education aims to improve students' meaningful learning and uses authentic situations in which scientific concepts are applied. The use of contexts may contribute to the learning of abstract concepts such as algorithms. The selection of appropriate contexts, however, is challenging for teachers. It is therefore interesting to examine whether students can play an active role in the conception of such contexts and how designing contexts may contribute to student learning. As a case study, we investigated students' design of contexts for learning algorithms in upper secondary education. We developed lessons in which students collaboratively designed contexts and then reflected individually on all contexts proposed. At the end of these lessons, students completed a learner report. The students' design of contexts provided a remarkably wide range of learning outcomes. Students not only reported to have learned more about the lesson topic (algorithmic concepts and the application of

these), but the learner reports also reflected learning about the process (learning with contexts, designing contexts, and collaboration). Our findings suggest that designing contexts contributes to active learning. The results of this study may serve as recommendations for future research concerning the role of students in designing contexts.

Keywords: computer science education; algorithms; context-based education; collaborative design of contexts

1. Introduction

A context-based approach to education can contribute to meaningful learning (Gilbert, 2006; Parchmann et al., 2006; Diethelm et al., 2011; Habig et al., 2018). Using real-world situations as contexts in which concepts are applied provides relevance and meaning to subject matter and supports the connection between subjects learned in school and the everyday life of students (Bennett, 2003; King, 2012). In practice however, finding these contexts is challenging for teachers (Di Fuccia & Ralle, 2016; Nijenhuis-Voogt et al., 2021), particularly when it comes to selecting one that is interesting for all students. Students do not all have the same interests, and some prefer a context that is connected to their everyday lives while others are engaged by a challenging or more unusual context (Habig et al., 2018). The examination of what happens when students play an active role in the selection or design of contexts might be one way to address their various needs and interests. This is especially relevant when considering the teaching and learning of fundamental computer science concepts, such as algorithms. Teaching algorithms and the analysis thereof is nontrivial (Dagiene & Jevsikova, 2012) and is challenging for teachers (Yadav et al., 2016).

The student perspective of context-based learning was investigated by Van Vorst and Aydogmus (2021), who analyzed which contexts students (aged 14 – 15 years) choose from a selection provided for learning chemistry. Nonetheless, little is known about how students would design contexts and the potential impact this may have on their learning. Finding new contexts in which a concept is relevant may contribute to student learning because the concepts must be recontextualized. Although the German project Informatik im Kontext (IniK) described this recontextualization as the last phase in a context-

based teaching unit (Diethelm et al., 2011), we did not find any empirical investigations that examine the learning outcome of this student activity.

The aim of this study is to elucidate how designing contexts may contribute to the student learning of algorithms. We therefore investigated the content and variety of student learning from the collaborative design of contexts for learning algorithms. Students in their last year of secondary education (aged 17 – 18 years) were asked to design contexts during their Computer Science (CS) class, specifically for the concepts of algorithms. Students designed these contexts collaboratively, then reflected individually on all the reported contexts. At the end of these lessons, students were asked to fill in a learner report (De Groot, 1980). Learner reports are known for their usefulness when examining different types of learning outcomes (for instance, cognitive and affective) (e.g., Henze et al., 2020). By analyzing learner reports, we examined the students' learning outcomes to investigate how designing and considering contexts can be used in computer science education and how it may contribute to student learning.

This paper is organized as follows. We first explain the background to the study and provide a conceptual framework on the design of contexts for a context-based education. Then we outline the method used in this study and describe the design of the lessons in which our research activities took place. We then present our findings which is followed by the discussion, including a statement of the potential implications. Finally, our conclusions are presented.

2. Conceptual Framework

2.1 Design of Contexts for Context-Based Education

Context-based education is characterized by the use of realistic contexts as a foundation for learning science (Gilbert, 2006; Taconis et al., 2016; Sevian et al., 2018). The aim of this approach is to provide relevance and meaning to the scientific concept (Parchmann et al., 2006; Bennett, 2016) to foster meaningful learning.

The selection of useful contexts that contribute to this meaningful learning appears to be challenging. Contexts are time- and place-dependent (Pilot & Bulte, 2006), which requires the frequent updating of context-based teaching materials. Di Fuccia and Ralle (2016) examined a context-based project in

Germany (Chemie im Kontext), describing how 12 high school teachers collaborated to design a context-based learning environment. In that study, the teachers were concerned whether the use of contexts would be detrimental to the teaching of science content and therefore focused on 'competence-oriented contexts' (p. 100). Moreover, Dierdorp et al. (2014) examined to what extent professional practices can be used as meaningful contexts to create curricular coherence for senior high school students (aged 16 to 17). Their results revealed that many students experienced professional practices as a meaningful context. In addition, Habig et al. (2018) analyzed the influence of context characteristics on the interest of students (in grade 7 to 9) in learning chemistry and concluded that it is important to consider the interests and experiences of the individual learners because the efficacy of a context may vary between students. Furthermore, in a study of CS teachers' views on contexts for teaching algorithms in upper secondary education, Nijenhuis-Voogt et al. (2021) concluded that it may be difficult to find effective contexts that are connected to students' everyday life but simultaneously appropriate for teaching and learning algorithmic concepts.

Previous research has examined the challenges for teachers and curriculum developers in designing contexts; however, few studies have investigated the students' perspective on the selection or design of contexts. Van Vorst and Aydogmus (2021) analyzed which types of contexts students (aged 14 – 15 years) preferred when working on a context-based chemistry task, asking them to choose one of six context-based learning materials. The students' reasons for choosing a context were also investigated. This study revealed interesting differences in students' context choices related to their gender, performance, and degree of interest in chemistry, leading Van Vorst and Aydogmus (2021) to conclude that 'one context does not fit all' and propose an informed justification of the selected contexts.

To further investigate the students' perspective on the selection of contexts, another option might be to give students a role in designing contexts. Miedema et al. (2023) investigated contextualization in the domain of database education for third year software engineering and information systems science students and asked students to design and implement an engaging relational database to examine factors that describe an engaging database domain. Asking students to come up with new contexts may yield contexts that are interesting to them, but in addition, the activity itself may contribute to student learning. Designing and considering contexts might contribute to student learning because it requires so-called recontextualization (see Figure 1). Recontextualization is defined by Van Oers (1998) as "contextualizing something in a new way" (p. 483). Some authors (Holbrook & Rannikmae, 2017; Ummels et al., 2015; Wierdsma et al., 2016) have described recontextualization as the transfer of learned concepts from one context to another.

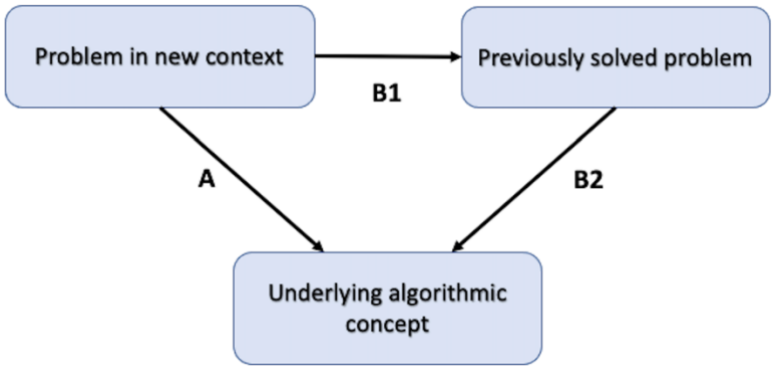

Figure 1.: Context-based learning and recontextualisation.

A teacher might use a context to explain a specific concept to ensure that students understand the concepts and see its relevance (line A in Figure 1). Subsequently, this concept can be recontextualized in new contexts (line B in Figure 1). These contexts may be offered by teachers (e.g., Wierdsma et al., 2016) or designed by students (as is the case in our study). Recontextualization in contexts designed by students was also proposed by Diethelm et al. (2011) as the final phase in a context-based teaching unit for IniK (the German context-based project for CS in lower secondary (grades 5 – 10) computer education): "In this phase, the transfer of the learned competencies and principles to other contexts should take place, so that students can answer the question for which other areas (contexts) the learnings

[das Gelernte] are relevant." (p. 103, translated from German). However, a systematic understanding of how this form of recontextualization contributes to student learning is still lacking.

2.2. Active Learning

Active learning is an important characteristic of context-based education (Gilbert, 2006). Active learning is consistent with a constructivist approach, which highlights the active role of the learner in building understanding. Pieters (2004) described how learners can 'take the lead' when they actively make meaning out of their activities and experiences in a discovery-learning environment. In addition, context-based education aims to provide students with a sense of ownership (Gilbert, 2006; Parchmann et al., 2006).

Prior research has shown how a sense of ownership positively influences student learning; for instance, by giving students a role in joint curriculum design where students and teachers collaborate and negotiate goals, content, methods, and assessment (Gross, 1997). In a study investigating the promotion of student autonomy and ownership, Chen et al. (2018) found that the creation of a comic play helped fifth-grade students in Taiwan in their learning of English as a foreign language because they could use their creativity and felt ownership for the presentation of their own script on a podium. In addition, Denny et al. (2015) examined students' design of a part of their learning environment and investigated the impact of inventing practice exercises in an introductory programming course at a university in Canada. Developing exercises may inspire students' creativity and provide them with a sense of ownership, and students who had participated in the exercise creation performed better on the exam. In the same study, Denny et al. (2015) examined how students perceived the process of inventing exercises and found that, according to students, inventing exercises had a positive impact on their learning of new concepts and enabled them to test their own understanding of the subject matter.

Similarly, in a study regarding student-generated quizzes, undergraduate students reported that developing questions improved their learning (Jones, 2019), empowering them to self-regulate their learning and take more ownership of their learning. Although student-regulated and self-directed learning is a valuable contribution to student learning (Assor et al., 2002; Schraw et al., 2006; Reeve,

2009), sometimes a higher degree of teacher-driven regulation may be worthwhile. Vermunt and Verloop (1999) elaborated on student-regulated learning and described a situation of 'constructive friction', such as students being asked to give examples but not being used to doing so. These frictions "may be necessary to make students willing to change and to stimulate them to develop skill in the use of learning and thinking activities they are not inclined to use on their own" (Vermunt & Verloop, 1999, p. 270). This challenge may promote new ways of learning and thinking in the students; however, there is a 'delicate balance' (Evans & Kozhevnikova, 2011) between sufficiently challenging students to try new ways of learning and thinking (constructive friction) and in overwhelming them, which may be destructive friction (Vermunt & Verloop, 1999). Prior studies have reported that creating effective situations of constructive friction may be motivating and stimulating, for instance in the context of learning calculus (Weurlander et al., 2017) or in the context of an integrated language-literature curriculum (Bloemert et al., 2019).

Together, these studies indicate the importance of students having an active role in, and a sense of ownership over, their learning, while also implying they benefit from constructive friction. In the present study, we assume that asking students to design contexts for learning algorithms may cause constructive friction (students have not done this before) and simultaneously may contribute to an active role and a sense of ownership. In the study presented here, we seek to identify how designing contexts may be used as a learning activity and what students learn from this process.

3. Aim of the Study

The aim of this study is to investigate how the design of contexts may contribute to student learning. As a case study, we investigated designing contexts for learning algorithms. We have translated our aim into the following main research question: What is the contribution of designing context for learning algorithms as perceived by students? This main requestion is divided into the following subquestions:

1. Which categories emerged based on the learner reports?
2. Which topics and learning outcomes have been addressed in the learner reports across the categories?

3. What contexts were preferred by students for searching and sorting algorithms and for routing algorithms?

4. Methods

4.1 Learning Design and Implementation Process

Our study was conducted in a VWO 6 class (last year of pre-university education) in the Netherlands. Secondary school students in the Netherlands can take CS classes as an elective in grades 10 – 12 (students aged 15–18 years). The recently revised CS curriculum (Barendsen et al., 2016) consists of a core curriculum and several elective themes, one of them being 'Algorithmics, computability and logic'. One of the aims of this elective theme is to teach students about the efficiency of algorithms and to explain the difference between exponential and polynomial complexity. Students are required to recognize and identify algorithms in various contexts.

We conducted our research in four lessons of the course regarding this algorithmics theme. During the lessons, several algorithms were discussed, including searching algorithms in unsorted and sorted datasets (linear and binary search), sorting algorithms, graph algorithms to find a shortest path (Dijkstra's algorithm) or a shortest round trip (Travelling Salesman). Various contexts were used to explain these algorithms; for example, Dijkstra's shortest path algorithm was explained in the context of a city map. In Table 1, an overview of the topics of the algorithmics lessons is given, with the contexts mentioned in the teaching material.

Table 1. Overview of the topics of the Algorithmics lessons.

Problem	Algorithm	Context
Searching	Linear search, binary search	Google, WhatsApp
Sorting	Selection sort, insertion sort	Contacts
Shortest path	Dijkstra's algorithm	Maps
Shortest route	Travelling Salesman	Package delivery service

For this study, we asked the students to design and consider contexts for the taught algorithms. We therefore developed two activities: 1) students were divided into groups and were asked to come up with new contexts and 2) the students were asked to individually reflect on all the developed contexts. These activities were performed after the lessons on searching and sorting algorithms, and then after the lessons on shortest path and shortest route algorithms (see Figure 2).

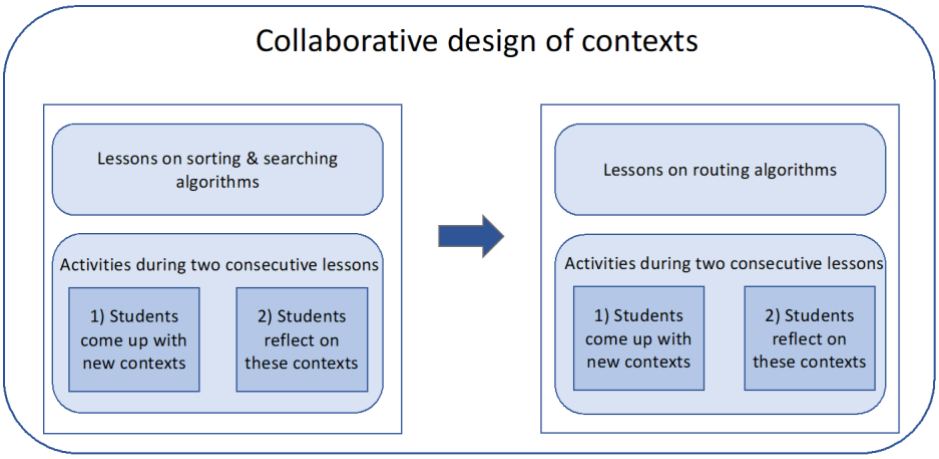

Figure 2.: Activities during lessons for collaborative design of contexts.

For the first activity, the students were randomly divided into groups of three or four and were given the following task:

> *"Think of a context in which you expect that searching and sorting algorithms can be applied* (second intervention: shortest path and shortest route algorithms). *For example, think of a concrete situation in which these algorithms are found. It might be a situation for which you would like to design or program something, or it might be something that you would like to understand better and where these algorithms could play a role. Try to come up with five very diverse contexts. Give a clear description of the context and explain why you have chosen it*."

During this activity, the aim was to diverge and to note anything that comes to mind.

The second activity took place during the subsequent lesson and was focused on converging. The students were handed a form with all contexts that came out of the first activity and were asked to evaluate these contexts individually. Contexts should be rated on a scale from 1 to 5 regarding the

following aspects: “to what extent do searching and/or searching algorithms appear in this context?” and “would you like to work with this context (is it interesting or challenging)?”

4.2 Participants

All 16 students of the VWO 6 class (last year of pre-university education, students aged 17–18) taught by the first author participated in this study. The teacher, who is also a PhD researcher, delivered the lessons in her own class. This study was carried out as part of this PhD research. The co-promotor of this PhD research was present during the lessons for reasons of objectivity and transparency. The class consisted of three female and thirteen male students. Class size and distribution of identified gender represents the general situation of CS classes in the Netherlands. Thus, we used ‘complete target population’ (all students in this class participated in the study) as a sampling strategy (Ravitch & Carl, 2015).

The students could be expected to have basic knowledge of the standard algorithms for which contexts had to be designed, given that the research activities took place after the lessons on these algorithms. The two consecutive lessons (for coming up with contexts and for reflecting on contexts, see Figure 2) were conducted on the same day, because CS lessons for this class (two 45-min lessons per week) were scheduled in the morning and the afternoon of the same day. After the lessons and research activities regarding searching and sorting algorithms, the students were taught about routing algorithms and participated in the research activities regarding this topic.

Our research proposal was approved by the local Ethics Committee and the data were collected, stored, and analyzed in accordance with the ethical research conduct.

4.3 Data Collection

The data used to examine how designing contexts for algorithmic concepts contributes to student learning consisted of the completed learner reports. The second author was present during all lessons to assist in collecting data.

A written learner report based on the format of De Groot (1980) was completed by the students at the end of the last research lesson. Learner reports are well suited to evaluation studies and are valuable for examining what the students themselves feel they have learned (e.g., Kesteren, 1993). Learner reports

have recently been used to examine the development of student learning about socio-scientific issues (Bayram-Jacobs et al., 2018; Henze et al., 2020), to investigate student attitude outcomes (Barendsen & Henze, 2015), and to examine the contributions of a course on teaching for creativity for student teachers (Oosterheert et al., 2020). These studies confirm that the use of learner reports yields rich data, especially if the learning results are not purely cognitive (Henze et al., 2020).

According to the format of De Groot (1980), the learning outcomes were reported in sentences such as "From the lessons 'contexts for algorithms' I have learned ...". These sentences are classified in two ways: 1) students report learning about 'the world' and about themselves; 2) students report learning from generalities (things that always happen) and learning from exceptions (new, surprising things). Through this categorization, the learner report consists of four types of sentences (see Table 2).

Table 2. Outline of learner report.

	Sentences to be completed
A	From the lessons 'contexts for algorithms', I have learned that . ..
B	From the lessons 'contexts for algorithms', I have learned that it is not true that ...
C	From the lessons 'contexts for algorithms', I have learned that I .. .
D	From the lessons 'contexts for algorithms', I have learned that it is not true that I ...

4.4 Data Analysis

A qualitative analysis was carried out to investigate what students reported to have learned from designing new contexts. We aimed to capture the different types of understanding within a cohort of students and to discover how students experienced this collaborative design. In this respect, our approach has elements in common with phenomenographic research (Marton, 1986; Tight, 2016).

As a first step in the analysis, repeated reading of the learner reports was carried out to develop an understanding of the data as a whole. During the next step, we classified the students' statements into broad categories that emerged from the data: students reported learning about the lesson topic

(algorithms and the application of algorithms) or about the process (learning with contexts and designing contexts). We then checked the data for the completeness of the categories. As a result of this step, we decided that we needed to add a fifth category namely 'collaboration' because it was mentioned several times in the data. To describe the data, we identified the distribution of reported sentences across the categories (see Table 3 in the Results section).

Following the labelling of the broad categories, we explored the details within each category. We therefore analyzed the content of the learning outcomes for these five categories and coded the learner reports using Atlas.ti qualitative data analysis software. Codes to describe the content of the reported learning outcomes were inductively developed using in vivo coding (Miles et al., 2014). The units of analysis were complete sentences in the learner reports.

This step of the data analysis revealed 14 learning outcomes, each related to one of the five content categories. For instance, a sentence in the learner report such as *I now have a general understanding of which different types of searching/sorting/routing algorithms there are*. was first labeled as a statement regarding 'learning about algorithms' and in the next step coded as 'functioning of algorithms'. In addition, the sentence *I have noticed that designing contexts requires quite a bit of creativity.* was first labeled with the content category 'learning about designing contexts' and then coded as 'designing contexts takes effort'. The categories and codes are used to organize the results section and are listed in Table 4.

To ensure the trustworthiness of our qualitative approach, we paid specific attention to the dependability of our study, an aspect that belongs to the criteria Lincoln and Guba (1985) defined to assess the quality and rigor of qualitative research. The first and second author (who were both present during the data collection) met almost every week to discuss the coding process to establish well-defined codes. The first author was the main coder. At different stages in the analysis process, the first and second author discussed parts of the coded transcripts to confirm interpretations. Therefore, we attempted to reach consensus rather than assess inter-rater reliability in a formal, quantitative sense (McDonald et al., 2019). Meetings with all authors (that took place every 6 weeks) were designed to discuss whether relevant data was missing or whether too much importance was attached to certain information, to ensure a

balanced data selection (Boeije, 2010). The third and fourth author often adopted a critical attitude toward the analysis to avoid biased interpretations.

5. Results

In this section, we present the results of the analysis to answer our main research question: What is the contribution of designing contexts for learning algorithms as perceived by students? We first give a clear overview of the data and describe what categories emerged based on the learner reports and how the reported sentences were distributed across the categories (Table 3). Furthermore, we describe the topics and learning outcomes according to the five categories of learning that were identified in the data. Finally, we address what contexts were preferred by students for searching and sorting algorithms and for routing algorithms.

5.1 Categories and Distribution in the Learner Reports

The analysis of the learner reports resulted in five categories: learning about algorithms, learning about the application of algorithms, learning about learning with contexts, learning about designing contexts, and learning about collaboration. As described in the Data Collection section, the learner report consisted of very open questions in which students were asked what they had learned (see Table 2). In the data as a whole, we found the five categories listed, but not all students reported having learned about all these categories. For example, Hugo wrote six sentences which have been classified into two categories (‘Learning about learning with contexts’ and ‘Learning about designing contexts’ see Table 3). That does not imply that Hugo did not learn about algorithms or the application of algorithms or about collaboration. It reflects that only learning about learning with contexts and learning about designing contexts came to mind when Hugo was asked what he had learned.

The distribution of sentences across these categories showed that almost all content categories were mentioned by many (9, 11 or 12 out of 16) students but sentences about ‘learning about collaboration’ were only found in the learner reports of four students. Furthermore, this analysis revealed that the students reported a wide range of learning outcomes: 75% of the students (12 out of 16) wrote sentences

about three or more content categories (see Table 3). We did not further analyze these results because of the small sample size (N = 16).

Table 3. Distribution of reported sentences across categories.

Student name (pseudonym)	Learning about algorithms	Learning about application of algorithms	Learning about learning with contexts	Learning about designing contexts	Learning about collaboration	Total number of sentences per student
Hugo	- *	-	5	1	-	6
Aiden	1	4	-	1	-	6
Otto	1	2	2	1	-	6
Roy	3	2	2	-	-	7
Tom	3	1	-	2	1	7
Elena	4	1	2	-	-	7
Chris	3	-	1	3	-	7
Frank	-	-	4	3	-	7
Leo	-	3	5	-	-	8
Robin	-	-	-	8	1	9
Kim	6	1	-	2	-	9
Alec	3	5	-	1	-	9
John	2	1	2	4	-	9
Jim	1	3	3	2	1	10
Carl	2	5	3	1	-	11
Demi	1	1	4	2	3	11
Total number of sentences per category	30	29	33	31	6	129

*: not coded in the data

5.2 Topics and Learner Outcomes in the Learner Reports

The analysis of what students reported to have learned revealed several topics and learner outcomes. In Table 4, the five categories are shown alongside the 14 identified learning outcomes. For each of the

learning outcomes, example quotes from students are given. The students' quotes were translated from Dutch into English.

The students reported having learned about **algorithms**. According to half of the participating students (8 out of 16), they learned about standard algorithms and how they work, which improved their conceptual understanding of algorithms. Seven students reported to have learned that there may be several algorithms that may be used to solve a problem and, in such a case, how to choose the best (or most efficient) algorithm. Three students specifically reported to have learned that an efficient algorithm for a shortest route does not yet exist. One student seemed to generalize this insight and reported to have learned that there is not always a good algorithm for a problem. Furthermore, the students reported several other insights that they learned from the lessons, for instance about the relationship between algorithms and coding or computers. Two students mentioned realizing that developing algorithms does not equal coding, and that algorithms are used outside of computers as well.

In addition, the students reported to have learned about **the application of algorithms**. Eight students described that they had learned that there are many applications for algorithms and that algorithms may be used in unexpected contexts. In the learner reports, three students noted that they now realized that certain algorithms were apparent in their daily life and had practical value. These students reported that their view of algorithms had changed; they were more aware of the broad applicability of algorithms and the *real applications* for algorithms.

Furthermore, the students described having learned about **learning with contexts**. Ten students reported that contexts contribute to learning and ensure a better understanding of algorithms; one of the students wrote down that lessons without context would not only be boring but also not understandable. According to six of the students, learning with contexts is appealing. These students reported that they enjoyed the lessons more when an interesting context was used, as a good context can make an assignment more fun. Not all students were enthusiastic about contexts, however; one preferred learning theory without contexts, and another student warned that contexts do not always contribute to learning because they may make a topic unnecessarily complex. By designing contexts, some students learned about the characteristics of useful contexts; five students wrote sentences in their learner report about

characteristics for contexts. These students reported for example that a context from the real world is more appealing, or that a challenging context is more interesting for some students. And one student commented that opinions on whether a context is interesting or not differ from person to person.

Students also described learning about **designing contexts**. Many (11 out of 16) students commented that it was not easy to come up with contexts. These students stated that it took effort and time, but in the end they were able to come up with contexts. Three students reported how they came up with contexts, such as writing down whatever came to mind or making associations based on the algorithm. In the learner reports, six students described how designing contexts contributed to their learning about algorithms and how they enjoyed this activity. They reported that designing contexts makes you feel connected to the topic to be learned. Context design was viewed as educational by these students because you are actively working on the design assignment and because you are only capable of designing contexts for standard algorithms when you understand these algorithms.

Finally, our analysis of the data revealed that students learned about **collaboration**. Only a few students reported learning outcomes that were related to this category (4 out of 16). They commented that collaborating supported the design of contexts, for instance because discussing algorithms and contexts with peers is regarded as exciting and challenging. One of these students was less enthusiastic about the collaboration however, stating that collaborating can be difficult and its success depends on the peers in your group. Depending on their relationships with the other group members, students might be encouraged or, on the contrary, discouraged to say what comes to mind when designing contexts.

Table 4. Categories and learning outcomes with example quotes.

Categories	Learning outcomes with example quotes
Learning about algorithms	Understanding the functioning of algorithms • I now have a general understanding of which different types of searching/sorting/routing algorithms there are. (Carl) Realization that multiple algorithms may exist for a problem; learned to compare and decide which is the best one

	• By having to think about it myself, I have noticed the differences between the searching algorithms and which algorithm would be preferred in certain circumstances. (John) • I have discovered that the different ways of sorting are not always the most efficient for every situation. (Kim) Awareness of limitations of (our knowledge about) algorithms • I now know that there is no efficient algorithm for the shortest roundtrip. (Chris)
Learning about application of algorithms	Discovery that numerous applications exist for algorithms, in many contexts • I have discovered that there are many applications for algorithms. (Otto) • I have learned that the route algorithm also appears in very unexpected contexts. (Leo) Recognition of the relationship between algorithms and daily life • I now recognize certain algorithms in my daily activities. (John) • I have discovered that algorithms are everywhere in the activities of our daily lives. (Alec) Broadened understanding of algorithms and their applicability • I have discovered that I use algorithms subconsciously more often than I expected. (Otto) • I have discovered that algorithms are not just used in computers. (Aidan)
Learning about learning with contexts	Realization that contexts contribute to learning

	• I now know that I learn efficiently with examples and situations that I recognize and that I can only then apply them to a new situation or think of one. (Elena) Discovery that learning with contexts is appealing • I have learned that a proper context ensures that you do the assignment not for your teacher, but for yourself. (Hugo) Awareness that contexts may be counterproductive • I now know that this way of learning does not help me to understand the material better; it's easier for me to learn a technique or theory without a context. (Chris) • I have learned that there should be a good balance between how interesting the context is and how clearly you can understand what it teaches you. Too much of one thing often comes at the expense of another. (Hugo) Knowledge of characteristics of useful contexts • I now know that I find more complex contexts more interesting as they are less obvious. (Frank) • I have learned that not everyone finds the same contexts interesting. Not only do teachers and students differ in opinion about whether a context is fun or not, there are also differences between the students. (Hugo)
Learning about designing contexts	Realization that designing contexts takes effort • I have learned that it is very difficult to design contexts for a topic that are well-suited to the material. (Robin) • I have noticed that designing contexts requires quite a bit of creativity. (Demi) Discovery that designing contexts is educational

	• I have noticed that you learn a lot from designing contexts as you are more involved with it yourself, and you have to know how it works. (Robin) • I have discovered that this way of learning appeals to me in principle. This is because you receive an explanation about how things fit together, and then you start designing yourself. (Tom)
Learning about collaboration	Discovery that collaboration challenges students and contributes to solving problems • I have learned that a particular problem may be resolved by talking about it with other people such as the group you are part of. This means there are multiple heads that can solve the problem. (Tom) • I have noticed that discussing algorithms and designing contexts in groups stimulates me. (Jim) Realization that collaborating with others can be difficult • I have learned that assessing other people's contexts is quite difficult as you often do not know what the underlying thought process is. (Demi)

5.3 Preferred Contexts for specific Algorithms

As described under *Learning Design and Implementation Process*, students were first asked to come up with new contexts. In the next lesson, the students were asked to individually reflect on all the developed contexts. These two activities were carried out after a lesson regarding searching and sorting algorithms, and then after a lesson on shortest path and shortest route algorithms. For the reflection activity, students rated to what extent an algorithm was clearly present in the context. In addition, they rated to what extent the context was appealing (interesting or challenging).

The following tables give an overview of the contexts that were rated as contexts with the 'clearest appearance of algorithm' or as 'most appealing' by the students, both for the searching and sorting

algorithms (Table 5) and for the routing algorithms (Table 6). It is worth noting that the students had a clear preference for contexts connected to their daily life. In addition, it reveals the 'cultural dependency' of contexts; for example, upper secondary students in other countries might not make frequent use of public transportation, and hence for them an app for public transportation may not be a context with the clearest appearance of a routing algorithm.

Table 5. Top three contexts for searching and sorting algorithms.

Criterion	Top three
Clearest appearance of searching or sorting algorithms	1. In the library or on their website, one can sort books by title, author or, for instance, by publisher. 2. Searching algorithms in a database, for instance at school: a database with data for all students. 3. Sorting contacts stored in your phone.
Most appealing	1. Sorting Netflix recommendations. 2. Searching for a criminal in a crowd using cameras and pictures of the criminal. 3. Spotify: sorting music and artists.

Table 6. Top three contexts for routing algorithms.

Criterion	Top three
Clearest appearance of routing algorithms	1. An app (e.g., 9292) that calculates the shortest route from one place to another using public transportation. 2. Google maps. 3. You would like to travel by rail through Europe; you know which cities you would like to visit and you want to spend as little time on the train as possible; what is the shortest route?
Most appealing	1. The Wikipedia game: go from a randomly selected article to another pre-selected target article, solely by clicking links within each article; arrive at the target article in the fewest clicks. 2. Selecting a new route when a train has been cancelled with as little traveling time as possible. 3. In an amusement park, you would like to see and do all rides and attractions with the least amount of walking; what is the shortest round trip?

6. Discussion

The main aim of this study was to explore how designing contexts for learning algorithms may contribute to student learning. We investigated learning outcomes after students were asked to design context for the concepts of algorithms. The most interesting finding was that the students' design of contexts provided a wide range of learning outcomes; students not only reported to have learned more about the lesson topic (learning algorithmic concepts and the application of these concepts), but also learning about the process itself (learning with contexts, designing contexts and collaboration). These different types of learning outcomes are in line with the work of Vermunt and Verloop (1999), who distinguished cognitive, affective, and metacognitive components of learning. Designing contexts contributed to changes in students' knowledge base (cognitive) and helped students to gain insight and

exert control over their learning (meta-cognitive). The affective aspect also emerged in the data, such as when students noted that contexts contributed to their motivation.

The analysis of the data indicated that the lessons in which students designed contexts for algorithmic concepts contributed to their learning about algorithms. Students reported to have learned about algorithms and about their efficiency. This result suggests that the learning outcomes overlap with the learning goals of the lesson series. The learning goals of these lessons also included recognizing and identifying algorithms in various contexts. The second part of the research activities (in which the students reflected on contexts) in particular may have contributed to this learning goal. Students were asked to reflect on all designed contexts and assess whether specific algorithms clearly appear in each of these contexts. Students' design of contexts and reflection on these contexts seems to align with the goals of a context-based approach that aims to contribute to meaningful learning (Gilbert, 2006; Parchmann et al., 2006).

Furthermore, it is interesting to observe the top three of contexts with clearest appearance of searching or sorting algorithms (Table 5) or routing algorithms (Table 6). These contexts are very suitable to use when teaching these algorithms, in fact, some of these contexts have been used in teaching materials (e.g., Google maps for routing algorithms). However, another context from this top-three is – to the knowledge of the students' teacher – not used in teaching material and consequently more a new context: "*You would like to travel by rail through Europe; you know which cities you would like to visit and you want to spend as little time on the train as possible; what is the shortest route?"* Although this context is still quite similar to the ones used in class before, it can be seen as a new context to which the learned algorithm is transferred (Diethelm et al., 2011). This recontextualization (Van Oers, 1998) applies even more to the contexts that are listed as 'most appealing' as they are less similar to the contexts that have been used in class (e.g., "*Searching for a criminal in a crowd using cameras and pictures of the criminal.*"). Another context from the top-three of most appealing contexts ('*Selecting a new route when a train has been cancelled with as little traveling time as possible*') reveals that the students who came up with this context connected the learned algorithm with their everyday life (Bennett, 2003), because

these students had experienced this themselves and had told their peer students and their teacher (the first author) about it recently.

In addition, the students reported learning outcomes regarding the application of algorithms. Some of the students described having learned that there are many applications for the algorithms they have learned, and that they now see how the learned concepts relate to their daily lives. This is clearly visible in the contexts that were generated by the students, e.g. the one about the cancelled train. Relating the learned concepts to their daily lives may contribute to a more 'profound form of learning' (Marton & Booth, 1997) because students view their daily lives differently as a result of class experiences. Learning about the application of algorithms does not completely equate to applying algorithms, but considering contexts where algorithms can be applied may be a first step in the application of algorithms. Designing and considering contexts may have contributed to the students' understanding of where algorithms can be applied.

Our findings showed that the participating students learned about the process: the students learned about learning with contexts, about designing contexts or about collaboration. Some of the students indicated that contexts contributed to learning, for example because they recognized the contexts or because a clear context provided a better understanding. Especially the contexts that were rated high on 'clear appearance' of the algorithm may have contributed to a better understanding.

In addition, some students discovered that learning with contexts is appealing, indicating that contexts may contribute to student motivation. This learning outcome is consistent with that of previous research, which found that contexts may provide meaning to subject matter (e.g., Bennett, 2016) and are effective for motivating students (Bennett et al., 2007). In this respect, it is interesting to note one of the students' awareness of the balance between the complexity of a context and its usefulness*: I have learned that there should be a good balance between how interesting the context is and how clearly you can understand what it teaches you. Too much of one thing often comes at the expense of another.* These findings are consistent with those of our previous study, in which interviewed CS teachers seemed concerned that the contexts for teaching algorithms can be too complex and may hinder the learning of

new concepts (Nijenhuis-Voogt et al., 2021). Although both studies were carried out in the context of teaching algorithms in CS, we might infer that these results could apply to other topics or subjects as well.

The results of the current study also showed that some students demonstrated knowledge of the characteristics of useful contexts; for instance, a context from everyday life is appealing. It is noteworthy that the students' remarks are in line with previous studies about a context-based approach to learning (Habig et al., 2018; Taconis et al., 2016) and add empirical evidence to these studies. Our findings also show diversity between students; students differ in whether they value a specific context as interesting. This is in line with the study by Van Vorst and Aydogmus (2021), who emphasized that one context 'does not fit all'. However, the use of a learning activity such as designing contexts may take these differences into account because students have the opportunity to design contexts that are appealing for themselves.

Furthermore, students described learning about designing contexts. Our results show that designing contexts is regarded as educational by students because they are actively involved. These results further support the idea of active learning, as demonstrated by Denny et al. (2015) in their research on students inventing exercises and as shown in the study of Jones (2019) about student-generated quizzes. Our findings also revealed that students needed to use their creativity to think of new contexts and that it took effort to find suitable contexts. Students described how they succeeded in coming up with contexts, even though it did not happen effortlessly. These findings corroborate the ideas of Vermunt and Verloop (1999), who described a situation of constructive friction that occurs when teachers challenge their students with activities they would not perform on their own, such as designing contexts. Based on our findings, we can speculate that students did not feel overwhelmed by too much friction (destructive friction) because they succeeded in designing contexts even though it took hard work.

Our findings regarding learning about collaboration indicated that designing contexts may contribute to an enriched understanding of collaboration with peers. Collaboration is regarded as one of the so-called 21st century skills. In the Framework of the Partnership for 21st Century Skills (P21, 2009), for example,

collaboration is referred to as one of the essential skills students must learn to be prepared for the future. Students must be able to work effectively in diverse teams and respect each other. Designing contexts may therefore enhance students' collaboration skills. With a limited number of students commenting on this theme (4 out of 16), caution must be applied, but the findings suggest interesting questions for further research.

As described in the conceptual framework, one of the aims of context-based education is to contribute to students' sense of ownership (Gilbert, 2006; Parchmann et al., 2006). Our findings suggest that asking students to design and consider contexts may yield this sense of control. One of the students reported: '*I have learned that a proper context ensures that you do the assignment not for your teacher, but for yourself*', indicating that he felt a sense of ownership. Although students experienced constructive friction caused by the teacher who controlled the difficult activity that took the students effort, this student simultaneously expressed a sense of ownership. However, these findings must be interpreted with caution because of the small sample size.

6.1 Implications, Limitations and Future Work

The present study suggests that designing contexts as a learning activity for students may be beneficial because designing contexts can provide a wide range of learning outcomes. Although the present study was a case study conducted in a CS class aimed at teaching and learning algorithms, we argue that our results provide insight into the use of designing contexts for teaching and learning CS in general and even for teaching and learning in a broader sense. The results of this study may serve as recommendations for further study, for instance regarding the role of constructive friction in a learning activity such as designing contexts.

In the lessons designed for our research, we focused on both the collaborative generation of contexts (a diverging activity) and the individual reflection on all resulting contexts (a converging activity). The converging part of the activities was new for students, as they were not used to reflecting on contexts. Although we did not specifically investigate whether the learning outcomes were related to either the diverging or the converging activity, we suggest that the reflection part in particular was instructive,

because students were asked to rate whether an algorithm was clearly apparent in the new context. This rating could only be carried out when a student knows the algorithm and is able to recognize that algorithm in a new context; thus, the student needs to recontextualize a concept (in this study: the algorithm). More research is required to investigate this preliminary implication further however, and to specifically examine the role of the reflection activity on the learning outcomes.

Students' individual reflection can also contribute to establishing whether a student-created context is valid. In the diverging activity, students might come up with all sorts of contexts but when students are individually asked "to what extent does the learning topic appear in this context", their answers may help to realize whether a context is valid for the specific learning topic. For teachers interested in using student-created contexts, we recommend using both the activities we designed for this research: the collaborative generation of contexts and the individual ratings of the generated contexts.

In common educational environments, it is difficult to account for the diverse interests and talents of all students in a class, but teacher awareness of these differences may contribute to a better anticipation and handling of this variation. By asking students to design contexts, teachers may gain insight into the interests and real everyday environment of students, such as our insight that public transportation plays an important role in the lives of the participating students.

Furthermore, the present study reveals that the investigation of learning outcomes using learner reports yields rich data. Collecting data with the use of learner reports is a very open way of asking for learner outcomes, more open than may be common in education. Although the current study is based on a small sample of participants, the findings raise interesting follow-up questions and give important starting points for future research.

Acknowledgements

We would like to thank all the students who participated in this study.

Funding

This research received funding from the Dutch Ministry of Education, Culture and Science under the Dudoc programme.

Ethics approval and consent to participate

Our research proposal was approved by the local Ethics Committee.

Competing interests

The authors declare that they have no competing interests.

References

Assor, A., Kaplan, H., & Roth, G. (2002). Choice is good, but relevance is excellent: Autonomy-enhancing and suppressing teacher behaviours predicting students' engagement in schoolwork. *British Journal of Educational Psychology*, *72*(2), 261–278. https://doi.org/10.1348/000709902158883

Bennett, J. (2003). *Teaching and learning science: A guide to recent research and its applications*. London: Continuum.

Barendsen, E., Grgurina, N., & Tolboom, J. (2016). A new informatics curriculum for secondary education in the Netherlands. In A. Brodnik & F. Tort (Eds.), *Informatics in schools: Improvement of informatics knowledge and perceptions* (pp. 105–117). Cham: Springer. https://doi.org/10.1007/978-3-319-46747-4_9

Barendsen, E., & Henze, I. (2015). *Teacher knowledge and students attitudes in context-based science education.* (Paper presented at NARST 2015, Chicago, IL)

Bayram-Jacobs, D., Henze, I., & Barendsen, E. (2018). The influence of ENGAGE materials on students' learning about socio-scientific issues. In *Proceedings of ESERA 2017*.

Bennett, J. (2016). Bringing science to life: Research evidence. In R. Taconis, P. den Brok, & A. Pilot (Eds.), *Teachers creating context-based learning environments in science* (pp. 21–39). Rotterdam, the Netherlands: SensePublishers. https://doi.org/10.1007/978-94-6300-684-2_2

Bennett, J., Lubben, F., & Hogarth, S. (2007). Bringing science to life: A synthesis of the research evidence on the effects of context-based and STS approaches to science teaching. *Science Education, 91*(3), 347–370. https://doi.org/10.1002/sce.20186

Bloemert, J., Paran, A., Jansen, E., & van de Grift, W. (2019). Students' perspective on the benefits of EFL literature education. *The Language Learning Journal, 47*(3), 371–384. https://doi.org/10.1080/09571736.2017.1298149

Boeije, H. (2010). *Analysis in qualitative research.* London: Sage.

Chen, G. D., Fan, C. Y., Chang, C. K., Chang, Y. H., & Chen, Y. H. (2018). Promoting autonomy and ownership in students studying English using digital comic performance-based learning. *Educational Technology Research and Development, 66*, 955–978. https://doi.org/10.1007/s11423-018-9597-7

Dagiene, V., & Jevsikova, T. (2012). Reasoning on the content of informatics education for beginners. *Social Sciences*, *78*(4), 84-90. https://doi.org/10.5755/j01.ss.78.4.3233

De Groot, A. D. (1980). Learner reports as a tool in the evaluation of psychotherapy. In W. de Moor & H. Wijngaarden (Eds.), *Psychotherapy, research and training* (pp. 177–182). Amsterdam, the Netherlands: Elsevier/North Holland Biomedical Press.

Denny, P., Cukierman, D., & Bhaskar, J. (2015). Measuring the effect of inventing practice exercises on learning in an introductory programming course. In *Proceedings of the 15th Koli Calling Conference on Computing Education Research* (pp. 13–22). New York, NY, USA: ACM. https://doi.org/10.1145/2828959.2828967

Di Fuccia, D.-S., & Ralle, B. (2016). Teachers in learning communities. In R. Taconis, P. den Brok, & A. Pilot (Eds.), *Teachers creating context-based learning environments in science* (pp. 89–101). Rotterdam, the Netherlands: SensePublishers. https://doi.org/10.1007/978-94-6300-684-2_6

Dierdorp, A., Bakker, A., van Maanen, J. A., & Eijkelhof, H. M. C. (2014). Meaningful statistics in professional practices as a bridge between mathematics and science: an evaluation of a design research project. *International Journal of STEM Education*, *1*(1), 9. https://doi.org/10.1186/s40594-014-0009-1

Diethelm, I., Koubek, J., & Witten, H. (2011). IniK - Informatik im Kontext: Entwicklungen, Merkmale und Perspektiven. *LOG IN, 31*(1), 97–104. https://doi.org/10.1007/BF03323736

Evans, C., & Kozhevnikova, M. (2011). Styles of practice: how learning is affected by students' and teachers' perceptions and beliefs, conceptions and approaches to learning. *Research Papers in Education*, *26*(2), 133–148. https://doi.org/10.1080/02671522.2011.561973

Gilbert, J. K. (2006). On the nature of "context" in chemical education. *International Journal of Science Education*, *28*(9), 957–976. https://doi.org/10.1080/09500690600702470

Gross, P. A. (1997). *Joint curriculum design: Facilitating learner ownership and active participation in secondary classrooms*. Mahwah, NJ, USA: Lawrence Erlbaum Associates.

Habig, S., Blankenburg, J., van Vorst, H., Fechner, S., Parchmann, I., & Sumfleth, E. (2018). Context characteristics and their effects on students' situational interest in chemistry. *International Journal of Science Education*, *40*(10), 1154–1175. https://doi.org/10.1080/09500693.2018.1470349

Henze, I., Bayram-Jacobs, D., Barendsen, E., & Platteel, T. (2020). Het bevorderen van burgerschapscompetenties in de natuurwetenschappelijke vakken met behulp van zelfgemaakt actueel en innovatief lesmateriaal. Retrieved from: https://repository.ubn.ru.nl/handle/2066/228005

Holbrook, J., & Rannikmae, M. (2017). Motivational science teaching using a context-based approach. In B. Akpan (Ed.), *Science education: a global perspective* (pp. 189–217). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-32351-0_10

Jones, J. A. (2019). Scaffolding self-regulated learning through student-generated quizzes. *Active Learning in Higher Education*, *20*(2), 115–126. https://doi.org/10.1177/1469787417735610

Kesteren, B. V. (1993). Applications of De Groot's "learner report": A tool to identify educational objectives and learning experiences. *Studies in Educational Evaluation*, *19*(1), 65–86. https://doi.org/10.1016/S0191-491X(05)80057-4

King, D. (2012). New perspectives on context-based chemistry education: Using a dialectical sociocultural approach to view teaching and learning. *Studies in Science Education*, *48*(1), 51–87. https://doi.org/10.1080/03057267.2012.655037

Lincoln, Y.S., & Guba, E. (1985). Establishing trustworthiness. In *Naturalistic inquiry* (chap. 11). Newbury Park, CA: Sage Publications.

Marton, F. (1986). Phenomenography – A research approach to investigating different understandings of reality. *Journal of Thought, 21*(3), 28–49.

Marton, F., & Booth, S. A. (1997). *Learning and awareness*. Mahwah, NJ, USA: Lawrence Erlbaum.

McDonald, N., Schoenebeck, S., & Forte, A. (2019). Reliability and inter-rater reliability in qualitative research: Norms and guidelines for CSCW and HCI Practice. *Proceedings of the ACM on Human-Computer Interaction, 3*(CSCW), 1–23. https://doi.org/10.1145/3359174

Miedema, D., Taipalus, T., & Aivaloglou, E. (2023). Students' perceptions on engaging database domains and structures. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2023)*, (pp. 122-128) New York, NY, USA: ACM. https://doi.org/10.1145/3545945.3569727

Miles, M. B., Huberman, A. M., & Saldana, J. (2014). *Qualitative data analysis: A method sourcebook*. Sage Publications.

Nijenhuis-Voogt, J., Bayram-Jacobs, D., Meijer, P. C., & Barendsen, E. (2021). Omnipresent yet elusive: Teachers' views on contexts for teaching algorithms in secondary education. *Computer Science Education, 31*(1), 30–59. https://doi.org/10.1080/08993408.2020.1783149

Oosterheert, I., Meijer, P. C., & van der Neut, I. (2020). Towards broader views on learning to teach: The case of a pedagogy for learning to teach for creativity. In D. R. Andron & G. Gruber (Eds.), *Education beyond crisis* (pp. 78–92). Dordrecht, the Netherlands: Brill Sense. https://doi.org/10.1163/9789004432048_005

P21. (2009). P21 Framework Definitions. *Framework for 21st Century Learning*. Retrieved from www.p21.org

Parchmann, I., Gräsel, C., Baer, A., Nentwig, P., Demuth, R., & Ralle, B. (2006). "Chemie im Kontext": A symbiotic implementation of a context-based teaching and learning approach. *International Journal of Science Education, 28*(9), 1041–1062. https://doi.org/10.1080/09500690600702512

Pieters, J. M. (2004). Designing artefacts for inquiry and collaboration when the learner takes the lead. *European Educational Research Journal, 3*(1), 77–100. https://doi.org/10.1080/09500690600702512

Pilot, A., & Bulte, A. M. W. (2006). The use of "contexts" as a challenge for the chemistry curriculum: Its successes and the need for further development and understanding. *International Journal of Science Education, 28*(9), 1087–1112. https://doi.org/10.1080/09500690600730737

Ravitch, S. M., & Carl, N. M. (2015). *Qualitative research: Bridging the conceptual, theoretical, and methodological*. Sage Publications.

Reeve, J. (2009). Why teachers adopt a controlling motivating style toward students and how they can become more autonomy supportive. *Educational Psychologist, 44*(3), 159–175. https://doi.org/10.1080/00461520903028990

Schraw, G., Crippen, K. J., & Hartley, K. (2006). Promoting self-regulation in science education: Metacognition as part of a broader perspective on learning. *Research in Science Education, 36*(1-2), 111–139. https://doi.org/10.1007/s11165-005-3917-8

Sevian, H., Dori, Y. J., & Parchmann, I. (2018). How does STEM context-based learning work: what we know and what we still do not know. *International Journal of Science Education, 40*(10), 1095–1107. https://doi.org/10.1080/09500693.2018.1470346

Taconis, R., Den Brok, P., & Pilot, A. (2016). *Teachers creating context-based learning environments in science*. Rotterdam, the Netherlands: SensePublishers. https://doi.org/10.1007/978-94-6300-684-2

Tight, M. (2016). Phenomenography: the development and application of an innovative research design in higher education research. *International Journal of Social Research Methodology*, 19(3), 319–338. https://doi.org/10.1080/13645579.2015.1010284

Ummels, M. H. J., Kamp, M. J. A., De Kroon, H., & Boersma, K. T. (2015). Promoting conceptual coherence within context-based biology education. *Science Education, 99*(5), 958–985. https://doi.org/10.1002/sce.21179

Van Oers, B. (1998). From context to contextualizing. *Learning and Instruction, 8*(6), 473–488. https://doi.org/10.1016/S0959-4752(98)00031-0

Van Vorst, H., & Aydogmus, H. (2021). One context fits all? – analysing students' context choice and their reasons for choosing a context-based task in chemistry education. *International Journal of Science Education, 43*(8), 1250-1272. https://doi.org/10.1080/09500693.2021.1908640

Vermunt, J. D., & Verloop, N. (1999). Congruence and friction between learning and teaching. *Learning and Instruction, 9*(3), 257–280. https://doi.org/10.1016/S0959-4752(98)00028-0

Weurlander, M., Cronhjort, M., & Filipsson, L. (2017). Engineering students' experiences of interactive teaching in calculus. *Higher Education Research & Development, 36*(4), 852–865. https://doi.org/10.1080/07294360.2016.1238880

Wierdsma, M., Knippels, M.-C., van Oers, B., & Boersma, K. (2016). Recontextualising cellular respiration in upper secondary biology education. Characteristics and practicability of a learning and teaching strategy. *Journal of Biological Education, 50*(3), 239–250. https://doi.org/10.1080/00219266.2015.1058842

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: understanding teacher experiences and challenges. *Computer Science Education, 26*(4), 235-254. https://doi.org/10.1080/08993408.2016.1257418

International Journal of Computer Science Education in Schools, November 2024, Vol. 7, No. 1
ISSN 2513-8359

Reverse Engineering in Robotics Classrooms: Boosting Creative Thinking and Problem Solving

Sevinç PARLAK
sevincparlak61@gmail.com
Computer and Instructional Technologies, Fatih Faculty of Education
Trabzon University, Trabzon

Neriman TOKEL
ntokel29@gmail.com
Computer and Instructional Technologies, Fatih Faculty of Education
Trabzon University, Trabzon

Ünal ÇAKIROĞLU
cakiroglu@trabzon.edu.tr
Computer and Instructional Technologies, Fatih Faculty of Education
Trabzon University, Trabzon

DOI: 10.21585/ijcses.v7i1.227

Abstract

This study explores the impact of robotics activities on the creativity and problem-solving performances of secondary school students. The participants consisted of 10 students from a computer science class at a secondary school. The robotics activities utilized Lego Ev3 kits and incorporated reverse engineering principles. Data were gathered using open-ended forms created to evaluate students' perspectives on creativity while engaging in tasks and their robotics problem-solving performances. The findings revealed that, most of the students demonstrated proficient skills, particularly in recognising problems, and creating alternatives, while their reasoning, applying the solution, and sharing skills were adequate. We hope this study will offer a valuable example of how to incorporate robotic activities within the reverse engineering approach.

Keywords: reverse engineering, robotics, creativity, problem-solving, secondary school students

1. Introduction

There is an increasing body of research highlighting the potential of educational robotics to enhance various problem-solving skills (Evripidou et al., 2020; Sun & Zhou, 2023). Educational robotics activities are implemented across all educational levels, from elementary to graduate, encompassing design, programming, application, and experimentation. Educators typically utilize robotics kits to construct robots and develop programs for specific tasks (Jung & Won, 2018). These activities can be structured as interventions, after-school programs, elective classes, or comprehensive course modules focused on robotics (Gubenko et al., 2021). For instance, Mblock is utilized for easy coding and control of robots. Lego Mindstorms allows students to design robots to enhance STEM skills and Lego Wedo 2.0 is another tool to develop children's dexterity. Mbot and Makey Makey are other tools for younger students to help them gain computational problem-solving experience. While mBot helps students to learn hands-on experience in the fields of graphical programming, electronics, and robotics; Makey Makey allows students to turn everyday objects into computer interfaces. Additionally, Vex IQ can be integrated with engineering challenges, making it a notable tool in educational robotics. Arduino is another educational robotics platform that requires basic knowledge of electricity and can be used to address simple real-life problems, such as watering plants or creating a door lock using various sensors. Students need to understand how to connect sensors, input-output devices, jumpers, and cables on a board, as well as use its programming language.

Robotics activities are used to enhance various skills and their potentials are closely related to how they are applied in the course process. In this sense, problem-solving and creativity skills are current key competencies that students need for their lives and professions in the future (Partnership for 21st Century Skills, 2009) and can also be enhanced through robotics. Thus, educators should help students keep up with these skills that is often referred to as twenty-first-century skills to meet challenges in the modern world. In a previous study, Costa and Fernandes (2005) applied robotics to space science, finding that students developed practical solutions to problems, which enhanced their critical thinking and logical reasoning skills. Similarly, Petre and Price (2004) demonstrated that robotics effectively helps primary and secondary school students grasp programming and engineering principles. Their findings indicated

that students were able to define the essential principles and concepts related to programming and engineering. Strawhacker and Bers (2015) reported how successful using the Lego WEDO robot set was in teaching basic programming concepts such as loop and decision-based to preschool students. In addition, Vatansever (2018) documented that using robots positively affected the problem-solving of fifth and sixth graders. Most of the aforementioned studies show that, in most of the robotics activities computational problems are given to students to be solved via educational robotics, and while acting in the robotics activities students gain various skills in the context of problem-solving and creativity.

The rest of the paper is structured as follows: Section 1.1 discusses creativity and problem-solving in educational robotics, while Section 1.2 covers descriptions and related work in the area of reverse engineering within educational robotics. We examine the relevant literature and identify research gaps in creativity and reverse engineering in computer science education. Following this, we present our research design and findings. Finally, we conclude with implications and suggest potential directions for future research.

1.1 Creativity and Robotics Problem-Solving

The application of educational robotics takes its roots in the constructionist approach (Danahy et al., 2014; Kafai & Resnick, 1996; Papert, 1981). While working in these constructionist situations, students think creatively and analyze the situation for problem-solving in real-world problems. Considering that creativity may be discussed in many broad senses, it is needed to understand the conceptualization of creativity in educational robotics contexts. Thus, general definitions, such as creativity being the generation of new ideas and innovative products (Kerr & Gagliardi, 2006), can be exhibited by engaging in robotics applications. In this process, creativity plays a crucial role in problem-solving, idea generation, conceptualization, theorization, and principle association (Azimpoor et al., 2017), helping to clarify the connection between robotics activities and creative outcomes. Prior research indicates that when given tasks designed to foster creativity in an educational setting, students are able to come up with innovative solutions and gain engineering competences by working through a range of computational issues. In addition to strengthening creativity, support, cooperation, and teamwork, educators seek to improve students' high-level thinking abilities, such as problem-solving, decision-

making, and scientific inquiry (Eteokleous-Grigoriou & Psomas, 2013).

The studies on educational robotics in classrooms provided some beneficial effects on creativity (Afari & Khine, 2017; Chetty, 2015). In one of the studies, Adams et al. (2010) interviewed engineering undergraduates who completed a voluntary Lego Mindstorms robotics module involving programming LEGO Mindstorms robot. The results showed most of the participants' perceived that their creative thinking skills had improved. Similarly, Cavas et al. (2012) introduced students to building and programming robots and investigated the effect of a LEGO Mindstorms robotics course on students' scientific creativity and found that students' creativity had increased after the program. Huei (2014) conducted a five-week program where students were introduced to a programming language for coding robots. Following the program, most students reported positive changes in their perspectives on creativity and problem-solving performances. In another study, Jagust et al. (2017) presented the results of workshops for gifted elementary students utilizing LEGO Mindstorms robotic sets. Although the authors did not conduct a psychometric assessment of creativity, their qualitative analysis concluded that the children demonstrated creative productivity. Eteokleous et al. (2018) evaluated the creativity of primary school students after a 36-week intervention and found a significant enhancement in their creative abilities.

However, it is typically challenging to identify creativity in studies that involve problem-solving. Four characteristics of creativity were proposed by researchers: fluency, flexibility, originality, and elaboration. Fluency in creativity is the ability to generate a large number of ideas quickly and effortlessly. In flexibility dimension, the student should be allowed to switch about their group assignments as needed. Originality is used to describe when students approach challenges from many points of view. The goal of the elaboration is to get the learner to value the specifics as they work toward the answer (Almeida et al. 2008; Kim, 2006).

Educational robotics provides experiential and situational learning for problem-solving real-world problems (Benitti, 2012). Students can be encouraged to think critically, analyze situations, and exercise creative thinking by engaging in. Previous studies show that working with educational robotics may allow students to create various solutions for computational robotics problems. However, it is generally

difficult for educators to construct learning environments to foster students' creativity. In this context, applying reverse engineering principles is used to support students in providing creative problem solutions in this study. Reverse engineering has been implemented in the field of mechanical components/assemblies, electronic components, and computer programs (Thayer, 2017).

In the educational robotics, students can examine existing robotic products to understand their construction, functionality, and programming. In this context, the potential of educational robotics for reverse engineering applications can provide innovative teaching methods for implementing robotics in education.

1.2 Reverse Engineering in Educational Robotics

Reverse engineering is defined as the process of creating a set of specifications for a piece of hardware by individuals other than the original designers, primarily through the analysis and measurement of a specimen or a collection of specimens. When using the reverse engineering approach, students examine existing systems to identify their components and the relationships among them (West et al., 2015). Thayer (2017) outlined the stages of reverse engineering as determining the design's purpose, observing its functionality, disassembling/distorting it, analyzing the components, and then reporting or redesigning based on the findings.

Reverse engineering helps students better understand the science behind design and the components of engineering design by offering them the opportunity to critically ask questions about design features (West et al., 2015). In this process, an existing product is examined and its detailed characteristics are sought to learn how the product is made and how it works. The reverse engineering process aims to reproduce an existing object by analyzing the dimensions, shape, and properties of the product or object (Batni et al., 2010). Thus, it is recognized as a powerful method for developing students' thinking skills (Dempere, 2009; Verner & Greenholts, 2016). It also enhances students' ability to systematically structure critical thinking (Griffin et al., 2012; Rogers-Chapman, 2014) and improves their logical thinking skills, thereby strengthening their analytical abilities (Klimek et al., 2011).

One of the prior studies emphasized the importance of reverse engineering in robotics education,

particularly in fostering creativity and improving learning performance, although it requires careful planning and design (Zhong et al., 2021). Another study matched the numerical data provided by the software interface, using a suitable symbolic model of the robot dynamics. The results showed that including training and validation tests with additional dynamic validation experiments that use the complete identified model and joint torque sensor data (Gaz et al., 2014). In previous studies; while the reverse engineering approach has been explored in various contexts, examples of its application in robotics education for primary and secondary schools are still limited.

Reverse engineering with Lego Ev3 might begin with the robot as a whole so that students can analyze its parts, pieces, software, and other components. They can also work on it by putting it back together from the parts, or they can provide software to make it better or change it. Students may be able to use their creativity in reverse engineering through the robotics applications that incorporate both software and hardware components. By activating their creativity, students have the opportunity to modify the functionality of existing products by adding new components, sensors, or codes. Thus, we hypothesize that utilizing this type of robots could serve as a host for more creative expression.

1.3. Aim of the study

Considering the significance of reverse engineering in robotics activities, we seek to gain practical insights into how this approach impacts creativity and problem-solving. Thus, the following problem guided to research study:

Do robotics activities focused on reverse engineering impact the exhibition of creativity and robotics problem-solving performance among secondary school students?

2. Methods

The study focused on the potential of robotics activities for creativity and problem-solving. Since these experiences can be gathered through qualitative data, an exploratory case study was adopted to generate insights that can inform further investigation, particularly an underexplored area of reverse engineering. Explanatory case studies provide explanatory information to examine situations in detail to answer the questions "why" and "how" the cases occurred (Fisher & Ziviani, 2004).

2.1 Participants

The participants included 10 sixth-grade students (3 boys and 7 girls) enrolled in a computer science class at a private secondary school. Informed consent for participation in the activities was obtained from their parents. A preliminary interview was carried out to select the participants purposefully to ensure that they had no prior knowledge about robotic activities. Students were identified as S1, S2, and so forth.

2.1 Process

Participants were divided into three groups to carry out the activities. Two computer science teachers facilitated the sessions, encouraging students to take active roles. All activities spanned two lesson hours. In the first 15 minutes, students were introduced to the Lego EV3 robot, after which they assembled its components. They were tasked with identifying the robot's purpose, understanding how it operates, and learning how it is constructed. To explore the functions of its parts, students disassembled the robot. They were then asked to design a solution for the given task. Throughout the activities, the teacher posed guiding questions to encourage student creativity. One of the researchers, who also served as the teacher, observed their behaviors during the tasks and helped them when needed. Four activities in the study are Carrying objects, Creating a bridge from objects, Moving the ship, and Unloading from the tower. The activities were associated with the steps of reverse engineering approach practically as presented in Table 1.

Table 1. Robotics Problems Regarding Reverse Engineering

Robotics Problem-Solving	**Reverse Engineering**
Recognizing the problem	Establishing the design objective
Articulates the problem in their own words.	Monitoring its functionality

Reasoning Employs reasoning (illustrating) in connection with the question or problem. Design solutions Engages in a design project to formulate a solution that adheres to particular design requirements and limitations. For example, students create a piece of the robot (ie carry an object) write codes for the task, and test it by running the robot.	Disassemble/deform Analyze
Problem Solving Addresses the problem by constructing and programming the model, making adjustments as necessary. Sharing Discusses the problem and proposed solution with the peers.	Report/Redesign

For example, in the first activity (Carrying Objects), students were tasked with transporting items. They were provided with a robot and asked to identify its purpose. During this phase, guiding questions prompted students to observe how the robot functioned. Next, students disassembled the robot to examine its parts and understand their functions. They were then instructed to design a robot suitable for the task, taking all previous stages into account. Students made modifications to both the code and design of the robot. Finally, they presented their designs to the class.

2.2 Data Collection Tools

2.3.1 Open Ended Questions Form

Questions for Robotics Problem-Solving

Open-ended questions were asked to understand thinking processes in robotics problem-solving after being reviewed by two computer science teachers as experts. Six open-ended questions included in the test were asked to determine the problem-solving performances regarding the robotics problems. Some example questions are included in Appendix 2.

Questions for Creativity

To explain the students' exhibition of their creativity, open-ended questions in the form regarding the reverse engineering approach was used. The questions were directed to the students when they were acting on the tasks. Some examples are presented in Appendix 3.

2.3 Data Analysis

The qualitative data on the exhibition of creativity was analysed through content analysis of students' perspectives. Themes and codes were developed based on indicators of creativity and the relationships observed in students' behaviors while engaging in tasks. Robotics problem-solving performances were assessed through a framework regarding problem-solving and scored by problem-solving rubric designed by the researchers.

2.4 Robotics Problem-Solving Rubric

Open Open-ended questions for robotics problem-solving were analysed by using the Robotics Problem-Solving Rubric. The rubric is created by adopting Polya's problem-solving steps. The criteria for the given problems are shown in Appendix 1. Example questions and evaluations are shown in Table 2.

Table 2. Example questions and evaluations about robotics problem-solving rubric

Robotics Problem Solving Steps	**Evaluation of Students' Statements**
Recognizing the problem	The problem was defined as follows: "How does the robot transport the object from its starting location back to the original point?" This student's statement was recognized as advanced, aligning with the advanced level in the rubric.

Reasoning	The statement, “If we design an additional part to prevent the object from falling and attach it to the robot, it can easily carry the object,” was assessed as a basic-level idea. This student’s response was deemed sufficient, corresponding to the sufficient level in the rubric.
Creating Alternatives	The robot’s ability to make the additional part that it can carry without dropping is considered a basic-level design. This design of the student was accepted as sufficient, coinciding with the sufficient level in the rubric.
Applying the solution	The student's inability to offer an alternative to the designed attachment to the robot to carry the object without dropping it coincided with the sufficient level in the rubric, and this design of the student was accepted as sufficient
Sharing	The students’ evaluations like “The robot was able to take the object from its location and move it to the starting point without any problems. The fact that the part is especially easy to attach and remove from the robot increases its functionality” was accepted as the advanced level.

The scoring was done by two different researchers who compared their scores. In cases where discrepancies arose, the researchers continued to evaluate until they reached a consensus and agreed on the final scores.

3. Findings

3.1. Students' Exhibitions of Creativity

The exhibition of students’ creativity were categorized into themes: reshaping, innovative ideas, leaving the current task, and detailing. Reshaping, one of the themes, focuses on the intention to alter the current design of the robot. Students' responses to the question" How can this robot be constructed differently?" are summarized in Table 3.

Table 3. Students' evaluations about "Reshaping"

Reshaping
Modifying the current *design* (n=3)
Generating alternative solutions by enhancing the current *design* (n=4)
Performing multiple tasks by creating additional components to *design* (n=4)
Creating new one while considering various variables of the *design* (n=4)
Achieving the solution through various approaches by utilizing different *code* blocks (n=6)
Completing the task more efficiently by employing different *codes* (n=2

Regarding the design, S1 stated: "If we change the design of the robot, we can design an additional arm", and S3 expressed: "A long arm can be added". Some other students pointed out the size of the code which was used to activate the robot. For example, S2 expressed that " I think if we use repetition blocks, we can solve more shortly", and S4 expressed that "We should utilize various code blocks for solution". Figure 1 shows a view of the design configuration in Activity 1.

Figure 1. A view from designing additional parts

Figure 1 illustrates students' behaviors as they create new components. The students indicated that while designing these parts, they aimed to enable them to perform multiple tasks. In this section, they created a new design by using various code blocks.

Another way of creativity is found in performing Innovative Ideas. The perspectives of students related to creativity are shown in Table 4.

Table 4. Students' evaluations about "Innovative ideas"

Innovative Ideas
Completing the task faster with different sensors(n=2) and proposing new project ideas (n=3)
Taking various functions into account (n=2)
Offering transactions that will facilitate the task (n=4)
Thinking alternative functions outside the robot's current role (n=3)

During their work, students were asked questions such as, "What would you like to change in this design?" "Can the designed part be made more functional?" and "Can this robot perform its task more efficiently? What would make that possible?" Some students responded with suggestions like, "We can do it in a shorter time using sensors," and "We can perform several tasks by assigning new tasks to the robot." These responses led to the emergence of innovative ideas.

A view from students exploring new solutions using sensors is shown in Figure 2.

Figure 2. A view from students exploring new solutions using sensors.

In the context of innovative ideas, students modified the codes related to the sensors, achieving the robot's final goal (the task outlined in Activity 2) by proposing new projects that incorporated different sensors attached to the robot. Another theme related to creativity was the Change of the Mission, which was further divided into two dimensions: coding and design. When asked, "Do you want to change the

task?" students' responses were categorized under the theme of mission change. The perspectives were presented in Table 4.

Table 4. Students' evaluations about "Leaving the current task"

Leaving the current task
Refusing the change because of an interest in design (n=3)
Refraining from job changes unless essential (n=3)
Focus on achievable tasks in codes (n=5)
Declining to change tasks due to interest in coding (n=2)
Declining a job change because of inadequate coding knowledge (n=3)

While some students declined to change the tasks, expressing, "No, I don't want to because I don't like to write codes," others agreed to move on to another part of the activity, stating, " No, I don't want to code; I'm not skilled at designing." which reflected their perceptions of coding tasks. Conversely, other students voiced their enthusiasm for design, saying, "I want to be a designer; it's more fun to design," and "I don't want to change my task; I want to design."

Some students were more focused on interacting with the robot and assembling the parts, showing less interest in coding. Their comments indicated a preference for creating new solutions by altering the robot's components and experimenting through trial and error. The evaluations of the students are illustrated in Figure 3.

Figure 3. A view from the coding and design tasks

Figure 3 shows that, the students who refused to change their tasks due to their interest in design continued to design tasks. A student working in the coding area accepted the task change and transitioned to the design area.

The last theme is detailing which includes detailing on coding and detailing on design. The evaluations of the students under the detailing theme are presented in Table 5.

Table 5. Students' evaluations about "Detailing"

Detailing
Attending to details in the solution process and *design* (n=4).
Utilizing the *design* space with precision (n=4)
Concentrating on the specifics of the *coding* area throughout the process (n=5)

Given the emphasis students placed on details, they were asked: "Is detailing important during the activities?" and "What specific details did you focus on when building the robot?" Students provided responses such as, "Yes, I paid attention to the details, especially in the codes, because even the smallest code error can prevent the robot from starting," and "I focused mainly on the codes; dealing with design isn't my responsibility."

Regarding design, students noted, "The detail of the attachment designed is very important because many features must be considered for the robot to function," and "I paid more attention to the details in the design because coding is ineffective without proper design." These expressions indicate that students recognized the significance of writing accurate code, assembling components, and viewing the robot as an integrated system. They understood that thinking through their tasks in detail would aid in successfully completing them. Students considering the detailing are shown in Figure 4.

Figure 4. A view from the students considering the detailing

Figure 4 shows that students discussed the reason in detail of the attachment and the importance of this attachment in the creation of the event.

Overall, the students' perspectives demonstrate that participating in robotics activities through the reverse engineering approach enhanced their creative problem-solving abilities. Two key components; design, which pertains to the hardware, and code, which refers to the programming blocks, were essential in showcasing their creative ideas and behaviors.

3.2. Students' Robotics Problem-solving Performances

Robotic problem-solving performances were assessed and rated by analysing and categorizing students' solutions. Their perspectives on these solutions were aligned with Polya's problem-solving steps, which were adapted for the robotics challenges. The level of students' in robotics problem-solving performances is shown in Figure 5.

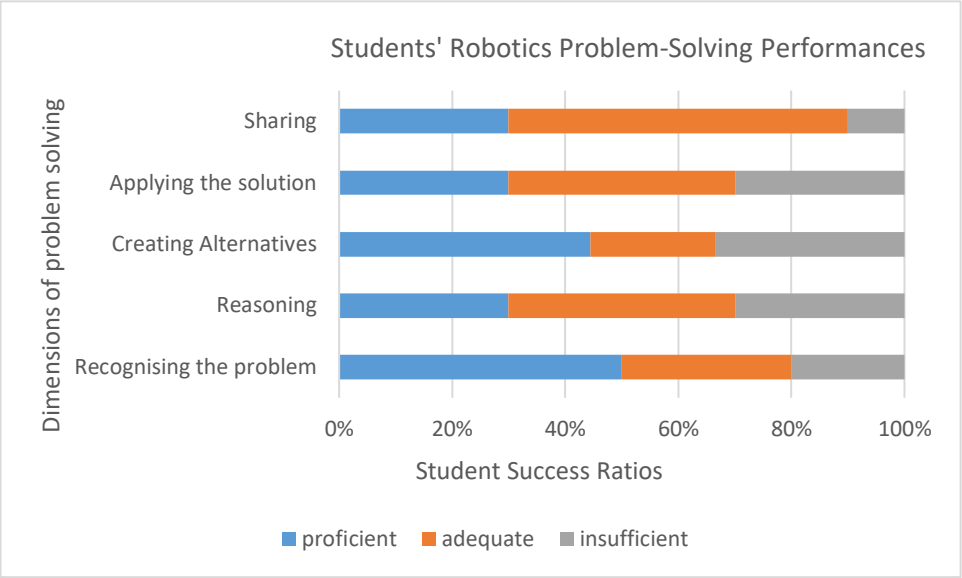

Figure 5. Students' robotics problem-solving performances

In the dimension of recognizing the problem, 5 out of 10 students participating in the research were at the "proficient" level, while 3 students were at the adequate level and 2 students were at the "insufficient" level. Regarding the reasoning, 3 out of 10 students participating in the research were at the "proficient" level, while 4 students were at the "adequate" level and 3 students were at the "insufficient" level. In addition, 4 out of 10 students participating were at the "proficient" level, 3 students were at the "adequate" level and 3 students were at the "insufficient" level. Moreover, in the sharing, 3 out of 10 students participating in the research were at the "proficient" level, while 6 students were at the "adequate" level.

Overall, the results indicate that engaging in reverse engineering positively impacted all components of problem-solving, with identifying the problem and designing solutions being particularly prominent skills during the implementation.

4. Discussion

This study investigated the impact of the reverse engineering approach in robotics activities in relation to creativity and robotics problem-solving. Initially, students built a basic robot using Lego Mindstorms sets, after which they were tasked with designing additional parts for specific challenges. It is believed that the parts created by the students contribute to the originality aspect of creativity. The students

generated various designs that successfully met the tasks. This production of diverse ideas and innovative solutions for each activity is thought to positively influence the fluency aspect of creativity. Zhong et al. (2021) also found that students who used reverse engineering did noticeably better than those in the project-based study group and had greater levels of creative self-efficacy, which is consistent with our findings.

The study evaluated students' robotics problem-solving performances on a number of different fronts. It was noted that they had improved in terms of comprehending the issue, making clear the work at hand, and figuring out how to resolve it. Students were given several reverse projects reflecting engineering to understand the purpose of the robots and the assigned tasks during the robotics activities. The majority of students were able to reason, solve the problems, and take into account other viewpoints. Students were urged to come up with various concepts for how the robot would operate in this situation. Because of the variety of ideas that the students present, it is thought that this stage effectively fosters reasoning.

Students created solutions to the problems given in the activities and came up with alternative ideas. It is thought that the students used their different ideas in the reasoning step to design solutions. Students generally did not struggle with the assigned problems and presented various ideas related to reverse engineering. The reverse engineering in the robotics activities enabled students to comprehend the problem, engage in reasoning, and develop solutions. Subsequently, the design activities offered students the opportunity to create new designs and code, taking into account all the stages of the process. Similarly; West et al. (2015) highlighted that employing the reverse engineering approach allows students to ask insightful questions about design elements, enhancing their understanding of the engineering and scientific principles involved in the design process.

Students presented their solutions with their group members after finishing the assigned assignments. For the most part, students met the requirements for sharing. The students were given opportunities to participate actively in group settings within the parameters of the activities. Students with various ideas are thought to generate a creative environment through discussion, and the presentation the students made to evaluate the design they created at the end of the session was deemed sufficient. Similar to the current study, reverse engineering was shown to improve academic performance, increase the self-

efficacy in programming, and have a favorable impact on their analytical and holistic thinking abilities when it comes to problem-solving (Abdüsselam, Turan-Güntepe & Durukan, 2022). Additionally, Taşçı and Şahin (2020) found that reverse engineering applications can enhance students' academic performance and problem-solving skills in science classes. Klimek et al. (2011) also explored the application of reverse engineering to teach scientific concepts, alongside engineering principles related to biomimetic robots. They reported that students gained both conceptual knowledge and practical mechanical engineering strategies through this approach.

Overall, the results supported the possibility of implementing reverse engineering activities in educational robotics. The beneficial effects on creativity and robotics problem-solving suggest that teaching problem-solving through robotics might be accomplished through reverse engineering. The activities in this study were modified to fit the phases involved in addressing problems. This could be one of the causes of the innovative solutions that have been developed.

This study is not exempt from limitations. The research was carried out with 10 students. Since the study is explanatory, a small sample provided to ensure analyzing the students' robotics problem-solving performances and creativity in detail. Since the study sample was small size, the effects with larger sample sizes should provide more sensitive results. In the activities, a limited number of robotic sets were used and the activities were carried out as group work as well as individual work. It should be noted that; concentrating on viewpoints gave us important information on how to use reverse engineering in educational robotics applications. Only the one type of the robots were taken into consideration while applying reverse engineering in this study. To learn about the building of robots and the possibilities of reverse engineering, alternative educational robots can be utilized.

5. Conclusion and Implementations

The study examined the impact of the reverse engineering approach in robotics activities on creativity and problem-solving skills. In terms of creativity, it can be concluded that reverse engineering activities can provide task-based flexibility. It has come beforehand to offer processes that will facilitate the task in the coding process. The reverse engineering approach facilitated to thinking of new functions of the

robots using different sensors. The ability to do more than one task by designing an additional part by paying attention to many variables came forward. The situation of solving in different ways by using different code blocks is a contribution of the approach. One can infer that reverse engineering activities provide reshaping in terms of originality and the students act in more detail in the coding area and using the design. Considering the stages of problem-solving; students demonstrated performances in recognizing the problem, reasoning, creating alternatives, applying the solution, and sharing their results. The findings revealed that the reverse engineering approach had a positive impact on the recognizing problem, where students needed to fully understand the task by illustrating the problem with various examples.

Ultimately, the results indicate that the robotic activities based on reverse engineering had a positive impact on students' robotics problem-solving performances. Both the findings and limitations offer potential pathways for future educational robotics activities that focus on creative problem-solving strategies. We hope that the insights from this study will facilitate the future integration of the reverse engineering approach in teaching other computer science topics.

References

Abdüsselam, M. S., Turan-Güntepe, E., & Durukan, Ü. G. (2022). Programming education in the frameworks of reverse engineering and theory of didactical situations. *Education and Information Technologies*, 27-5, 6513-6532.

Adams, J. C. (2010). Scratching middle schoolers' creative itch. In Proceedings of the 41st ACM Technical Symposium on Computer Science Education (pp. 356-360).

Afari, E., & Khine, M. S. (2017). Robotics as an educational tool: Impact of Lego Mindstorms. *International Journal of Information and Education Technology*, 7(6), 437-442.

Almeida, L. S., L. P. Prieto, M. Ferrando, E. Oliveira, and C. Ferrándiz. 2008. "Torrance Test of Creative Thinking: The Question of its Construct Validity." *Thinking Skills and Creativity*, 3 (1): 53–58. doi:10.1016/j.tsc.2008.03.003.

Batni, S., Jain, M. L., & Tiwari, A. (2010). Reverse engineering: a brief review. *International Journal on Emerging Technologies*, 1 (2), 73-76.

Cavas, B., Kesercioglu, T., Holbrook, J., Rannikmae, M., Ozdogru, E., & Gokler, F. (2012). The effects of robotics club on the student's performance on science process & scientific creativity skills and perceptions on robots, human and society. In Proceedings of 3rd International Workshop Teaching Robotics, Teaching with Robotics Integrating Robotics in School Curriculum (Vol. 40, p. 50).,

Costa, M. F., & Fernandes, J. F. (2005). Robots at school. The *Eurobotice project. Science and Technology*, 1, 2.

Dempere, L. A. (2009). Reverse engineering as an educational tool for sustainability. In *2009 IEEE International Symposium on Sustainable Systems and Technology* (pp. 1-3). IEEE.

Evripidou, S., Georgiou, K., Doitsidis, L., Amanatiadis, A. A., Zinonos, Z., & Chatzichristofis, S. A. (2020). Educational robotics: Platforms, competitions, and expected learning outcomes. *IEEE Access*, 8, 219534–219562. https://doi.org/10.1109/ACCESS.2020.3042555

Eteokleous-Grigoriou, N., & Psomas, C. (2013). Integrating robotics as an interdisciplinary-educational tool in primary education. In *Society for Information Technology & Teacher Education International Conference* (pp. 3877-3881). Association for the Advancement of Computing in Education (AACE).

Eteokleous, N., Nisiforou, E., Christodoulou, C., Liu, L., & Gibson, D. (2018). Fostering children's creative thinking: A pioneer educational robotics curriculum. *Research Highlights in Technology and Teachers Education*, 89-98.

Fisher, I., & Ziviani, J. (2004). Explanatory case studies: Implications and applications for clinical research. *Australian Occupational Therapy Journal*, 51(4), 185-191.

Gaz, C., Flacco, F., & De Luca, A. (2014). Identifying the dynamic model used by the KUKA LWR: A reverse engineering approach. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1386-1392). IEEE.

Griffith, A. L. (2010). Persistence of women and minorities in STEM field majors: Is it the school that matters? *Economics of Education Review*, 29(6), 911-922.

Gubenko, A., Kirsch, C., Smilek, J. N., Lubart, T., & Houssemand, C. (2021). Educational Robotics and Robot Creativity: An Interdisciplinary Dialogue. Frontiers in Robotics and AI, 8, 178.

Huei, Y. C. (2014). Benefits and introduction to Python programming for fresh more students using inexpensive robots. In *2014 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)* (pp. 12-17). IEEE.

Jung, S. E., & Won, E. S. (2018). Systematic review of research trends in robotics education for young children. *Sustainability*, 10(4), 905.

Kerr, B., & Gagliardi, A. (2006). Measuring creativity in research and practice. Arizona State University.

Kafai, Y. B., & Resnick, M. (1996). Constructionism in Practice: Designing, Thinking, and Learning in a Digital World. Mahwah, NJ: Lawrence Erlbaum.

Kim, K. H. 2006. Can We Trust Creativity Tests? A Review of the Torrance Tests of Creative Thinking (TTCT). Creativity Research Journal, 18 (1): 3–14.

Klimek, I., Keltika, M., & Jakab, F. (2011). Reverse engineering as an education tool in computer science. In *2011 9th International Conference on Emerging Elearning Technologies and Applications* (ICETA) (pp. 123-126). IEEE.

Papert, S. (1981). Mindstorms: Children, Computers, and Powerful Ideas. UK: Harvester Press.

Petre, M., & Price, B. (2004). Using robotics to motivate 'back door' learning. *Education and Information Technologies*, 9(2), 147-158.

Rogers-Chapman, M. F. (2014). Accessing STEM-focused education: Factors that contribute to the opportunity to attend STEM high schools across the United States. Education and Urban Society, 46(6), 716-737.

Sun, L., & Zhou, D. (2023). Effective instruction conditions for educational robotics to develop programming ability of K-12 students: A meta-analysis. *Journal of Computer Assisted Learning*, 39(2), 380-398.

Strawhacker, A., & Bers, M. U. (2015). "I want my robot to look for food": Comparing Kindergartner's programming comprehension using tangible, graphic, and hybrid user interfaces. *International Journal of Technology and Design Education*, 25(3), 293-319.

Thayer, K. (2017). How does reverse engineering work. Retrieved February, 6, 2021.

West, A. B., Sickel, A. J., & Cribbs, J. D. (2015). The science of solubility: Using reverse engineering to brew a perfect cup of coffee. *Science Activities*, 52(3), 65-73.

Vatansever, Ö. (2018). Examining the Effects of Using Scratch Programming on 5th and 6th Graders' Problem Solvıng Skills (Unpublished Master Dissertation, Bursa Uludag University, Turkey).

Verner, I., & Greenholts, M. (2016). Teacher education to analyze and design systems through reverse engineering. In International Conference EduRobotics 2016 (pp. 122-132). Springer, Cham.

Zhong, B., Kang, S., & Zhan, Z. (2021). Investigating the effect of reverse engineering pedagogy in K- 12 robotics education. *Computer Applications in Engineering Education*, 29(5), 1097-1111.

Appendix 1. Robotics Problem Solving Rubric (RPS-Rubric)

Robotics Problem- Solving dimensions	Criteria
Recognizing the problem	Can s/he understand the given scenario and explain it in his/her own words?
Reasoning	Does it offer alternative ideas to find solutions?
Creating alternatives	Can s/he implement his/her solutions ideas?
Applying the solution	Can s/he create new projects by making changes in the design s/he applies?
Sharing	Can s/he share the design s/he created with his/her classmates?

Appendix 2. Questions about Problem Solving

Robotics Problem-Solving

How can this design be done differently? Can this robot run using different codes? Can this designed piece be made more functional? What kind of innovations does the design and code part of the redesigned robot include compared to the first designed robot? Can you introduce the redesigned robot to your classmates?

Appendix 3. Questions about Creativity

Creativity

Can you explain the given scenario in your own words? Does it offer alternative ideas to reach solutions? Can you implement your ideas?, Can you create new projects by making changes in the design you applied? Is its design/coding successful or not? Can you share the design you created with your classmates?