Education and Information Technologies (2022) 27:10731–10756
https://doi.org/10.1007/s10639-022-11055-y

Whether to flip Extreme Apprenticeship: which is more effective in programming instruction?

Sinan Hopcan[1] · Elif Polat[1] · Ebru Albayrak[2]

Received: 31 January 2022 / Accepted: 8 April 2022 / Published online: 20 April 2022

Abstract
Programming knowledge is more important than ever in the digital world. However, teaching programming can be challenging, especially with novice learners. Considerable research has been conducted into the most effective methods for teaching programming. Extreme apprenticeship, a variation of cognitive apprenticeship, is a method that has been used in teaching programming at university level in recent years. Because this method focuses particularly on completing lots of exercises with coaching and guidance, it may solve many problems related to learning programming. Flipped learning can be useful for student preparedness and providing sufficient theoretical knowledge at the beginning of the course. This study compares the applications of the extreme apprenticeship method, flipped extreme apprenticeship, and traditional classroom, analyzing them at the university level in terms of their effects on academic achievement and engagement coupled with gender differences. The findings of the study indicate that the extreme apprenticeship and flipped extreme apprenticeship instructional methods improve academic achievement and student engagement in introductory programming more than the traditional method. The results of the research point to important directions for the development of the extreme apprenticeship method in programming instruction and provide a guide for instructors.

Keywords Cognitive apprenticeship · Extreme apprenticeship · Flipped classroom · Teaching programming

✉ Ebru Albayrak
ealbayrak@sakarya.edu.tr

[1] Computer Education and Instructional Technology Department, Istanbul University-Cerrahpasa, Istanbul, Turkey

[2] Computer Education and Instructional Technology Department, Sakarya University, Sakarya, Turkey

1 Introduction

In the knowledge era in which we currently live, rapid scientific and technological advances have made programming abilities essential for a skilled workforce and have emphasized the need for education in such skills (Campe et al., 2020; Witherspoon et al., 2016). People may need to use their programming skills to be more successful in their future work, putting more importance on the training they receive. Introductory programming courses have therefore become increasingly important over the last few years and have become a focus of researchers' attention (Giannakos et al., 2014a; Lahtinen et al., 2005; Nikula et al., 2011).

Programming is a multi-step process that involves analysis, strategy development, algorithm development, and code writing (Falloon, 2016; Hwang et al., 2012). To master programming, learners must learn both syntax and how to set up many logical structures, such as variables, loops, arrays, and recursions (Korhonen & Malmi, 2000). According to research, programming is an area in which students fail most (Hanks et al., 2011; Robins et al., 2003) and face disappointment (Bravo et al., 2005). Novice learners, in particular, may suffer from the process of learning programming skills (Koulouri et al., 2014).

Investigations of novice learners' learning methods found that students had significant difficulties in their courses because they tended to write codes line by line rather than establishing a logical structure (Robins et al., 2003). Many methodologies have been used in the literature to make programming teaching more effective in order to overcome this situation. Some of these methodologies are executing robotic applications (Witherspoon et al., 2016), pair programming (Lee, 2011), flipped learning (Zha et al., 2020), demonstration (Bean et al., 2015), collaborative learning (Rodríguez et al., 2017; Williams et al., 2003), and learning by creating games or stories (Kafai & Burke, 2013). However, the traditional method is still used by universities in introductory courses for programming. Rather than developing novice learners' problem-solving skills, many introductory programming courses teach them the syntax and meaning of programming (Iqbal Malik, 2016; Vihavainen et al., 2011a). This format includes lectures, take-home assignments, and sometimes demonstration sessions where exercises are shown as models (Vihavainen et al., 2011a).

Introduction to programming courses need to be redesigned and instructional methods based on a constructivist understanding should be adopted so that novice learners in particular can learn and internalize the logic of programming in greater depth based on their previous experience. For this, teaching with guidance, putting the student at the center of learning, will be beneficial. However, the nature of exercises and guidance is critical in this regard. According to Bruhn and Burton (2003), traditional lectures do not provide students with practical knowledge and students fail to understand how to apply programming concepts in their assignments. Given these disadvantages, unguided or minimally guided problem-solving exercises can be intimidating and confusing for inexperienced students (Bruhn & Burton, 2003). In fact, educational psychologists argue that minimal guidance is less effective, particularly for novice learners, in challenging subjects such as programming due to humans' cognitive architecture (Kirschner et al., 2006). With minimal guidance, students may be unable to complete an exercise, increasing the dropout rate and feelings of inadequacy (Vihavainen et al., 2011a). Furthermore, exercises that are not too difficult can give learners a higher quality learning experience by increasing their self-confidence and belief in their ability to solve problems (Wiedenbeck, 2005). Educators can improve their students' performance by teaching them strategies to put together pieces of code rather than having them memorize programming theory, as this helps them learn the syntactic and semantic structures of language (Spohrer & Soloway, 1986).

In general, a programming teaching approach should primarily focus on providing adequate support for novice learners (Kirschner et al., 2006). Researchers who design environments for teaching programming to novice learners have been drawn to the extreme apprenticeship (XA) method, which has been used in educational environments for the past ten years. Flipped learning is another method that can provide support to novice learners in terms of preparing them in advance for lessons. So far, however, the XA environment has not been applied in combination with flipped learning in the literature. This study adds to the field's innovation by testing the effects of these two environments on programming instruction.

2 Theoretical background

The theoretical background for the current discussion is structured through a review of the literature. Research has proven that student participation is weak in traditional methods, often due to the dullness of the content, especially when applied in disciplines with complex concepts and plenty of theoretical topics (Tang et al., 2020). In this environment, learners also cannot develop their high-level skills in programming due to the lack of practical knowledge and sufficient feedback (Bruhn & Burton, 2003; Carbonaro, 2019). Therefore, there is a need for environments that will increase learner engagement beyond the traditional to gain learning experience by doing (Yıldız Durak, 2018). In addition, because out-of-school experiences are limited to reading resources shared by the instructor, learners are often not willing to advance their learning processes. Because of all these shortcomings, the method is not as effective as the flipped (Taşpolat et al., 2021) and XA methods (Rämö et al., 2015) and can mean students lose their way in complex and challenging content such as programming (Hopcan et al., 2022). Studies that compare the traditional method with flipped (Amresh et al., 2013; Chen et al., 2014; Souza & Rodrigues, 2015) and XA methods (Plonka et al., 2015; Rämö et al., 2015; Vihavainen et al., 2011b) also found that the traditional approach did not contribute as much as others in essential factors such as engagement and achievement.

This article examined the effects of XA, flipped XA, and traditional methods relative to each other. In XA, plenty of supervision is given to learners at first and support is gradually withdrawn as the process progresses (Vihavainen et al., 2011a). Also, in this approach, less time is given to theoretical lectures than in the traditional method, giving more time for application. Flipped XA is an environment where these processes are reversed and contains the applications of XA in the classroom. Here, there is pre-lesson preparation in which learners benefit from videos prepared by the instructor. The traditional method, on the other hand, is a more teacher-centered approach where lectures are predominateand few exercises are provided. The following section explains XA and f lipped XA and presents how these two methods go beyond the traditional method. Research on the intertwined relationship between methods and learner achievement, engagement, and gender differences are described.

2.1 Extreme apprenticeship

Extreme apprenticeship, based on the cognitive apprenticeship approach, is a new methodology (Del Fatto et al., 2016) first used in introduction to programming courses at the University of Helsinki in 2011 (Vihavainen et al., 2011a). Cognitive apprenticeship is essentially an approach in which skills are acquired under the supervision of a subject matter expert who has prior practice and experience, with an emphasis on the learning the process rather than outcome (Solitro et al., 2016). Accordingly, cognitive apprenticeship enables effective cognitive development to take place under the supervision of an expert in learning environments that necessitate the use of metacognitive abilities (Plonka et al., 2015). With the potential to take programminginstruction to the next level, XA entails organizing programming instruction in such a way that it can more effectively address instruction, respond to needs, and deal with problems (Rämö et al., 2015). This method ensures that novice learners attain programming skills by completing a series of small exercises under the supervision of an expert (Del Fatto et al., 2016). These exercises are initially very simple. Support is gradually reduced as the learner progresses, allowing them to improve their cognitive processes on their own and eventually become an expert in the subject (Lee, 2020). Learners get plenty of feedback on their progress during exercises (Vihavainen et al., 2012); although the instructor does not give learners the answer but rather manages the scaffolding, process by providing hints (Vihavainen et al., 2011b). For example, instead of giving the answers directly to the learners, providing a concrete example of the solution enables them to discover the specific answer themselves. With these small hints, learners can advance their problem-solving skills. According to Sinha and Kapur (2021), it is important for learners to produce their own solutions, a process that helps them achieve higher order thinking. In this way, XA allows students to go beyond memorization and gain a deeper understanding of concepts (Rämö et al., 2015).

According to one research conducted on the XA method in higher education, learners' achievement rates did not decrease, they felt confident about learning, and the process was satisfactory even though their workload increased (Hautala et al., 2012). In another study, XA enabled learners to have a high level of self-efficacy and gain positive experiences (Lahdenperä et al., 2019). When XA was tried in programming courses, learners were found to perform satisfactorily (Pasini et al., 2016). Keijonen, Kurhila, and Vihavainen (2013) tried the XA method in their study and found that the learners were successful. Vihavainen et al. (2011b) found that in the programming course they taught using XA through a MOOC, learners felt that they had a quality learning experience.

2.2 Flipped XA

Flipped learning is an instructional method that promotes XA by encouraging students to prepare before class and actively participate in class with the knowledge they have gained (Rämö et al., 2015). In this method, learners work on exercises in a classroom setting and study passive course content in their own environment to achieve high-level learning (Sarawagi, 2013). Among these preparations are watching subject-related videos, studying shared resources, and doing subject-related homework prior to the lesson. Any questions that come to mind while students work individually are brought to the lesson and discussed (Amresh et al., 2013). As a result, it is possible to ensure that students perform mental exercises at a higher level.

It is critical that materials in the flipped environment are interesting and that the student receives feedback after watching videos to ensure quality of learning (Yıldız Durak, 2018). The role of the educator shifts from the role of the mere instructor to the role of a mentor. (King, 1993). It is important, in the constructivist understanding, that the instructor ensures that learners find answers on their own rather than providing a direct solution. In XA, too, after initially providing a large amount of support, learners become expert in what they do as the instructor gradually withdraws support while at the same time learners' progress from simple to complex via a master-apprentice relationship. Programming instruction requires long periods conducting a large number of exercises. In order to address the excessive amount of time required, this study involved the design and implementation of a flipped XA environment in which flipped learning was blended with the XA method. According to research conducted in higher education, flipped learning has a positive impact on programming instruction. Amresh et al. (2013) investigated the effects of the flipped learning environment and the traditional environment on the programming instruction of university students and discovered that the flipped environment resulted in higher achievement. Souza and Rodrigues (2015) found not only higher achievement but also higher self-efficacy among learners, while Chen et al. (2014) found that learners were more satisfied, motivated, and successful.

2.3 Engagement

According to research in the field, there is a significant dropout rate in programming instruction (McKinney & Denton, 2004). In this regard, learners' engagement levels as well as their achievement indicators should be closely monitored. Students' learning processes can be influenced by whether or not they attend school, complete their homework, and carry out the necessary preparation at home. Pattanaphanchai (2019) and Chen et al. (2014) found a significant relationship between achievement and engagement in the flipped environment. Maher, Latulipe, Lipford, and Rorrer (2015) used flipped learning and paired programming, as well as scaffolding. They noted that the students enjoyed and engaged well in this environment. In general, research has shown that when flipped learning is used, achievement and engagement increase. When the XA environment is combined with flipped learning, it is expected that students will arrive prepared for class, allowing for a reduction in the amount of theoretical teaching and the completion of a large number of targeted exercises, making this a beneficial approach. This is because novice learners require plenty of resources and individualized instruction (Lahtinen et al., 2005). In addition, learners are motivated to prepare for class, thus learning in their own time and at their own speed. However, the XA environment has not previously been studied in combination with flipped learning. This study is therefore the first to compare the effects of this process with both the XA environment and the traditional environment.

2.4 Gender differences

Gender differences in programming instruction have been documented and should not be overlooked. Kay (2006) came to the conclusion that male perceptions and interests in programming skills differ from females' due to their prior experiences. Furthermore, some studies have found that male students are more successful and engaged more in programming than female students, while others found no difference (Alvarado et al., 2017; McDowell et al., 2006; Tyler & Yessenbayeva, 2018). According to the literature, female students tend to have a lower rate of completion of programming courses (Black, 2007). These differences in programming instruction may be due to female students not interacting with computers as much as male students in their early years (Kay, 2006; Loftsson et al., 2019). Because of this, any new method for programming instruction should be assessed in terms of gender differences. The ultimate goal of this research was to create the most successful learning environment possible for students to gain valuable programming experience. Accordingly, the XA model was tried both alone and with a flipped classroom approach in an introduction to programming course, taking into account the gender factor in terms of achievement and engagement, and the effects of both learning environments compared to the traditional method.

For this purpose, answers to the following research question were sought: Is there any significant difference in the engagement and achievement of students in terms of programming instruction method and gender?

3 Method

This study applied extreme apprenticeship (XA), a variant of cognitive apprenticeship, in the flipped classroom and traditional classroom to teach the basics of programming at the university level. The study followed the matching-only posttest-only control group design as one of the quasi-experimental designs. The study followed a quasi-experimental design. Experimental design is defined as research designs that explore cause-effect relationships between variables (Fraenkel et al., 2012). The purpose of quasi-experimental design is the same as experimental design. The difference between them is that in quasi-experimental design, control and experimental groups are chosen not by chance but by measurement (Fraenkel et al., 2012). No random assignment was made in the selection of experimental and control groups and the prior performance test of the groups were checked for being equal in terms of academic achievement, which was the dependent variable of the research. The instructional strategies, XA, flipped XA, and traditional, were considered independent variables, and achievement and engagement were considered dependent variables. The procedure was carried out over a period of eight weeks. In both experimental and control groups, lessons were conducted by the two researchers, who have eight years of experience in their field. In order to prevent factors arising due to instructor differences affecting the research and threatening internal validity, the two instructors followed previously determined procedures and met on a weekly basis. A prior performance test was applied to ensure that the participants had similar prerequisite knowledge. Figure 1 demonstrates model of the study implementation.

The XA classroom involved short lectures, exercises, course book, and homework. The flipped XA classroom included pre-lesson videos, a brief review in a lecture, exercises, course book, and homework, while the traditional classroom included lectures, one exercise per lesson, and homework (see Figs. 2, 3 and 4). As previously mentioned, XA contained a lot of exercises. Lecturing was kept shorter than would be

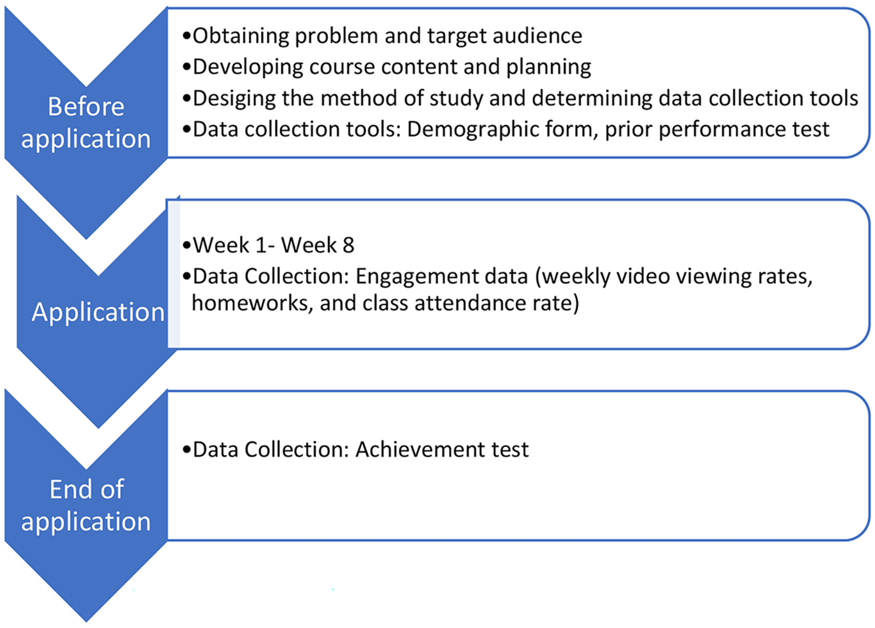
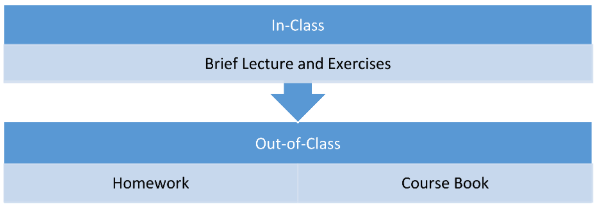

Before application
•Obtaining problem and target audience
•Developing course content and planning
•Desiging the method of study and determining data collection tools
•Data collection tools: Demographic form, prior performance test
Application
•Week 1- Week 8
•Data Collection: Engagement data (weekly video viewing rates, homeworks, and class attendance rate)
End of application
•Data Collection: Achievement test

Fig. 1 Model of Study Implementation

Fig. 2 XA Model



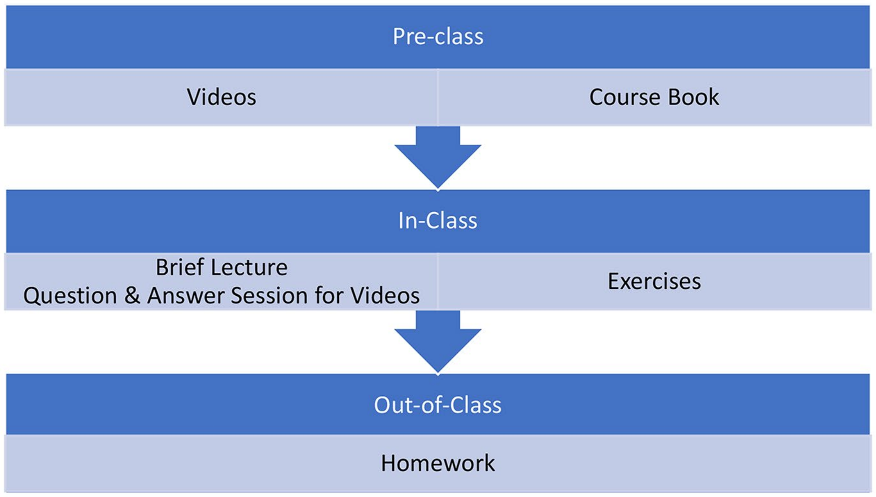
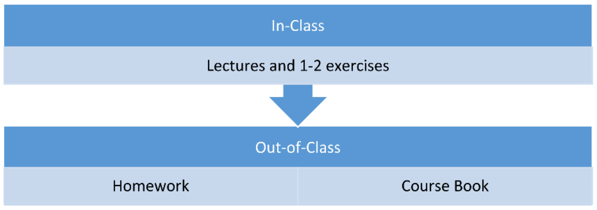


Fig. 3 Flipped XA Model

Fig. 4 Traditional Classroom Model



the case in the traditional classroom, where because of the greater element of lecturing, the number of exercises was low. For example, only minimal exercises could be done in the traditional class, while more exercises were done in the XA.

Since exercises are a vital part of this process, a number of exercises were prepared to cover each topic. In addition, the exercises were prepared in relation to the students' departments in order to better hold their attention. Examples of exercises are given in the Appendix A (see Appendices). The instructors provided a plenty of guidance to the students during the first exercises. As the student progress, the guidance was gradually reduced to have students construct a deeper understanding. Scaffolding was conducted by instructors and mentors. Mentors were all active and eager to help. Student teaching assistants, or mentors, scaffolded students in carrying out the exercises, monitoring as they walked around the laboratory. They provided continuous feedback in XA and flipped XA, guiding the students to the answers rather than directly providing solutions as instructors did. In addition, both instructors and mentors naturally guided the practices in the laboratory in the traditional classroom. There was traditional guidance as it is a hands-on course. Therefore, there were four or five mentors in the laboratory to each of the three groups. Communications of mentors and instructors were carried out with the instant messaging program WhatsApp. During the lesson, those with questions and answers relating to the exercises wrote to the group and received immediate feedback.

Table 1 The Extreme Apprenticeship Approach Used in the Study

Approach	Explanation
Avoiding long lectures	Long lectures were avoided throughout the course. Lectures were given only to enable students to complete the exercises.
Exercises related to the lectures	Exercises related to the lessons taught were included.
Starting early (starting exercises from the first week)	After the first lesson, the students began the exercises.
Constant help in labs	While the students were doing the exercises in the laboratory, the instructor and volunteer mentors, who were experienced with and knowledgeable about the programs, guided the students.
Small goals/Sense of achievement	The exercises were designed from easy to difficult to give students a sense of achievement. They were also divided into parts leading to the final destination.
Exercises are mandatory	Since the exercises were the main instrument of the course, the students were required to complete the exercises.
Plenty of practice with lots of repetition	As many exercise as possible were completed. Some of them were repetitive.
Clear guidance	Clear guidelines were given for the exercises.

Table 2 Course Content

Content	Pre-class Video Activities
General introduction	General introduction and how to open a project in MIT App Inventor.
Mathematical operations	Increasing the number one by one as the button is clicked. How to download and run the emulator.
Variables	Variables, changing the text of a label with the text of a textbox. Horizontal Arrangement object.
Conditionals (if-else)	Conditionals (if-else), visibility of labels.
Conditionals (if-else)	Facebook login page application. TableArrangement, image, passwordTextbox, AND-OR logical operation, Notifier.
List-Array	Write the city whose license plate is entered on the screen. List.
Loop	Writing the list elements to the screen. Loop (two application).
Functions	Calculation game based on random operations and numbers (math and function).

The extreme apprenticeship approach was used in the study for developing and delivering the course content (Vihavainen et al., 2011b). The approach used is presented in detail in Table 1.

The course content, lecture videos, in-class exercises, and out-of-class homework were developed by instructors with eight years of experience in programming instruction. While preparing the course content, the literature and practice area were examined (see Table 2).

3.1 Participants

The participants were 124 freshmen students enrolled in the Information Technologies (IT) course, an undergraduate program of the faculty of education at a state university in Istanbul. Istanbul is Turkey's most densely populated and metropolitan city. It was therefore considered appropriate to choose a state university based in Istanbul to ensure as much generalizability as possible. (See the conclusion for limitations to this.) Generally, research has been conducted on a single case (one faculty of education), whereas this current study provides a deeper and richer understanding in examining a situation (Turnbull et al., 2021). The students were from four different departments: Social Studies Teaching, Elementary Education Teaching, Preschool Teaching, and Turkish Language Teaching, which were randomly assigned to the three groups of the study. The students were of similar socio-economic level, had similar university entry scores, and were from similar ethnic backgrounds.

Table 3 demonstrates information of study participants. Of the participants, 76.61% ($n=94$) were female and 24.19% ($n=30$) were male. The number of participants in the experimental groups was equivalent, with 43 (34.68%) in the XA group, 44 (35.48%) in the f lipped XA group, and 37 (30.65%) in the traditional group.

3.2 Instruments

In this study, demographic form, the engagement data, a prior performance test, and a multiple-choice exam were employed to recruit data. Ethics approval was obtained before commencing the study. The study used a prior performance test and a multiple-choice exam as an achievement test to measure participants' level of programming knowledge. Since the number of correct answers in the prior performance test was very low, internal consistency analysis could not be performed. Sample items/questions from the instruments can be seen in Appendix B (see Appendices).

Table 3 Information of Study Participants

Gender	XA		Flipped XA				Traditional		Total	
	Classroom Teaching		Pre-school Teaching		Turkish Language Teaching		Social Studies Teaching			
	N	Per.	N	Per.	N	Per.	N	Per.	N	Per.
Female	35	28.23%	20	16.13%	14	11.29%	25	20.16%	94	76.61%
Male	8	6.45%	3	2.42%	7	5.65%	12	9.68%	30	24.19%
Total	43	34.68%	44		35.48%		37	30.65%	124	100%

Demographic form This form included six questions regarding gender, age, and computer knowledge. The form was developed by the researcher.

Prior performance test Since students from different departments participated in the study, the Basic Programming Prior Performance test was applied to reveal the students' basic programming and algorithm pre-knowledge. It consists of 17 questions. The opinions of two educational technology experts were obtained for the test.

Multiple-choice exam as achievement test The achievement test was developed by the researcher in order to measure the level of students' learning in basic programming. It consists of 20 multi-choice questions. Expert opinions on the questions were obtained from five information technology teachers and five educational technology experts. Revisions were made in line with the experts' opinions. Three questions with low item discrimination index (0.11, 0.14, and 0.17) were removed from the test. The item discrimination indexes of the remaining questions varied between 0.23 and 0.77, and the item difficulty indexes varied between 0.21 and 0.91. As a result of the KR-20 test, the reliability score was calculated as 0.80. As a result, 17 questions were asked.

The engagement data This score was found by recording students' attendance in face-to-face lessons, video watching rates, and homework completion rates. In the flipped classroom group, the engagement was the average of homework rate, video watching rate and class attendance rate; in the other two groups, it was the average of homework rate and class attendance rate.

3.3 Data Collection and Analysis

The study continued for eight weeks for three hours a week, teaching basic programming using the programming training tool MIT App Inventor (http://appinventor.mit.edu). MIT App inventor, one of the block-based visual programming tools, is an easy tool that can be learned in a short time. For this reason, it has been chosen as a learning tool for novice learners in this study (Polat & Hopcan, 2019). The experimental groups were XA ($N=43$) and flipped XA ($N=44$); the control group consisted of students who received traditional education ($N=37$). The researchers took some precautions in order to avoid issues affecting internal validity and generalizability (external validity). For increasing internal validity, a prior performance test was applied to ensure that the participants had similar prerequisite knowledge. It was therefore assumed that both groups were equal. To ensure that equality was maintained throughout the study, the researchers checked that the participating students did not attend a similar course for the duration of the intervention. Each group was presented with an equal number of resources and time. In order to prevent instructor differences affecting the research and threatening internal validity, the two instructors followed previously determined procedures and met to discuss them on a weekly basis. In addition, in order to ensure generalizability (external validity), the researchers tried to obtain a representative sample.

In the first lesson, the students were informed about the purpose of the lesson and the study, the content of the lesson, and the platforms of the lesson. The participants were informed about ethical principles, the purpose of the study, and that participation was voluntary. Google Classroom was used in all groups as a learning management platform and MIT App Inventor was used as a programming learning tool. In the flipped groups, the Edpuzzle platform was used for videos before the lesson. In Edpuzzle, multiple choice, fill-in-the-blank, matching type questions etc. were added to each video to check whether the students had watched it or not and to reinforce the topic. During the first lesson, students were asked to register on these platforms. The duration of the videos was about 15–16 min. In order to control the duration of intervention in the study, lesson time was kept the same for all three groups. Since there are pre-videos in the flipped classroom, the normal lesson time was shortened by 15 min and the lesson time was equalized with the other two groups.

SPSS 21 was used to analyze the data. Descriptive statistics was employed. Two way MANOVA were used to investigate effects of three independent variables on two dependent variables. The alpha level for data analysis was set at 0.05.

4 Results

The prior performance test scores presented in Table 4 show a normal distribution in the total. In the groups, the students were generally not successful in the prior performance test, and the average score of each group was low. This proves that the students did not have any prior knowledge about basic programming.

4.1 Two way MANOVA

A Two-Way MANOVA test was performed to determine the joint effect of two independent variables (gender and group) on two dependent variables (achievement score and engagement score). Table 5 presents the sample size, mean, and standard deviation values of each analysis group. As can be seen in Table 5, the highest engagement was in the flipped XA group (*M*=91.89, *SD*=10.24), the engagement scores of female students were higher than male students (Total *M*=86.87, *SD*=13.06), the highest achievement score was in the XA group (Total *M*=77.84, *SD*=15.97) and the female students' score (Total *M*=66.08, *SD*=20.64) in the groups other than in the flipped XA was higher than the male students' score (Total *M*=58.63, *SD*=21.15).

Table 4 Descriptive Information of the Prior Performance Test

	N	Min.	Max.	M	SD	Skewness	Kurtosis
XA	43	0	6	3.02	1.71	-0.10	-0.60
Flipped XA	44	0	6	3.32	1.91	0.02	-1.21
Traditional	37	0	6	2.16	1.36	1.01	0.94
Total	124	0	6	2.87	1.75	0.29	-0.87

Table 5 Descriptive Statistics

Dependent Variable	Category	Gender	Mean	SD	N
Engagement	XA	female	87	10.81	35
		male	75.17	14.1	8
		Total	84.8	12.22	43
	Flipped XA	female	92.06	9.97	34
		male	91.29	11.67	10
		Total	91.89	10.24	44
	Traditional	female	79.61	16.29	25
		male	78.34	18.47	12
		Total	79.19	16.78	37
	Total	female	86.87	13.06	94
		male	81.81	16.37	30
		Total	85.64	14.03	124
Achievement	XA	female	79.83	14.85	35
		male	69.12	18.8	8
		Total	77.84	15.97	43
	Flipped XA	female	66.44	16.09	34
		male	70.59	12.71	10
		Total	67.38	15.35	44
	Traditional	female	46.35	17.34	25
		male	41.67	17.28	12
		Total	44.83	17.22	37
	Total	female	66.08	20.64	94
		male	58.63	21.15	30
		Total	64.28	20.92	124

4.1.1 Assumptions

Since the skewness and kurtosis values of the dependent variables (achievement score: -0.36, -0.83; engagement score: -1.14, 0.81) were between −2 and +2, it can be said that they showed a normal distribution. The Mahalanobis distance value was measured as 11.37 (Pallant, 2005) (which should be less than 13.82 when there are two variables), and a multivariate normality assumption was provided. For the assumption of a linear relationship between each pair of the dependent variables across each level of the independent variables, scatter plots were examined and generally, a linear relationship was provided. The correlation between the two dependent variables was examined and found to be 0.32. Since it is larger than 0.2 (Pallant, 2005), relation assumption is provided, and since it is less than 0.9 (Pallant, 2005), multicollinearity assumption is provided. Looking at the Box's Test of Equality of Covariance Matrices results, it can be assumed that the covariance matrices of the dependent variables are equal across groups. Scores on the combination of dependent variables do not differ by either gender or interaction of gender and group.

As for the differences in gender sample sizes, providing the assumption of the multivariate homogeneity of variance-covariance matrices according to our Box's M test result provides evidence that sample size is acceptable for two-way MANOVA analysis (Van Huynh et al., 2018). Also, according to Hair et al. (2009), the cell sizes for MANOVA should be larger than the number of dependent variables. There are two independent variables in our analysis (engagement and achievement). Therefore, the number of samples per cell should be at least 3. As shown in Table 5, the lowest sample number is in the XA-male cell (n = 8). Since it was larger than 2, the number of samples was considered sufficient. In addition, because the number of samples per cell differs, Scheffe (Barnette & McLean, 2005) was used as a post hoc test.

4.1.2 Multivariate Test results

Because all assumptions were met, Wilks' Lambda values were checked. Looking at the multivariate test results in Table 6, the combination of achievement and engagement was not equal for all groups ($\lambda = 0.59$, F (4, 234) = 17.48, $p < 0.01$). A large effect size was observed ($\eta 2 = 0.23$). According to gender ($\lambda = 0.97$, F (2, 117) = 1.65, $p > 0.05$) and the joint effect of group and gender ($\lambda = 0.95$, F (4, 234) = 1.33, $p > 0.05$), there was no evidence of a significant difference between the combination of achievement and engagement scores.

Table 6 MANOVA Results

Effect	Λ	F	Hypothesis df	Error df	p	η^2
Intercept	0.03	2085.91	2	117	0.00	0.97
Group	0.59	17.48	4	234	0.00	0.23
Gender	0.97	1.65	2	117	0.20	0.03
group * gender	0.95	1.33	4	234	0.26	0.02

Table 7 Tests of Between-subjects Effects

Source	Dependent Variable	Type III Sum of Squares	df	Mean Square	F	P	η^2
Corrected Model	engagement	4213.68	5	842.73	4.97	0.00	0.17
	achievement	23381.06	5	4676.21	18.11	0.00	0.43
Intercept	engagement	623882.17	1	623882.17	3681.97	0.00	0.97
	achievement	344247.97	1	344247.97	1332.92	0.00	0.92
Group	engagement	2862.31	2	1431.15	8.45	0.00	0.13
	achievement	15824.98	2	7912.49	30.64	0.00	0.34
Gender	engagement	473.41	1	473.41	2.79	0.10	0.02
	achievement	311.39	1	311.39	1.21	0.27	0.01
group * gender	engagement	539.66	2	269.83	1.59	0.21	0.03
	achievement	802.40	2	401.20	1.55	0.22	0.03
Error	engagement	19994.18	118	169.44			
	achievement	30475.47	118	258.27			
Total	engagement	933742.82	124				
	achievement	566207.99	124				
Corrected Total	engagement	24207.86	123				
	achievement	53856.54	123				

4.1.3 Tests of between-subjects effects

Table 7 shows tests of between-subjects effects. The ANOVA results were significant and both achievement (F (2,118) = 30.64, $p < 0.01$) and engagement (F (2,118) = 8.45, $p < 0.01$) scores differed significantly between the groups. The achievement effect size was large (η2 = 0.34), and the engagement effect size was almost large (η2 = 0.13). There was no evidence of a significant difference in either dependent variable according to gender.

Since the number of individuals in each group differed, the Scheffe test was performed as post-hoc (Barnette & McLean, 2005). According to Table 8, Scheffe test results, there was a significant difference in achievement scores between XA and traditional in favor of XA, between XA and flipped XA in favor of XA, and between

Table 8 Multiple Comparisons

Dependent Variable	(I) Category	(J) Category	Mean Difference (I-J)	Std. Error	Sig.
Engagement	XA	Traditional	5.61	2.92	0.16
		Flipped XA	-7.09	2.79	0.04
	Flipped XA	XA	7.09	2.79	0.04
		Traditional	12.69	2.90	0.00
	Traditional	XA	-5.61	2.92	0.16
		Flipped XA	-12.69	2.90	0.00
Achievement	XA	Traditional	33.01	3.60	0.00
		Flipped XA	10.46	3.45	0.01
	Flipped XA	XA	-10.46	3.45	0.01
		Traditional	22.55	3.58	0.00
	Traditional	XA	-33.01	3.60	0.00
		Flipped XA	-22.55	3.58	0.00

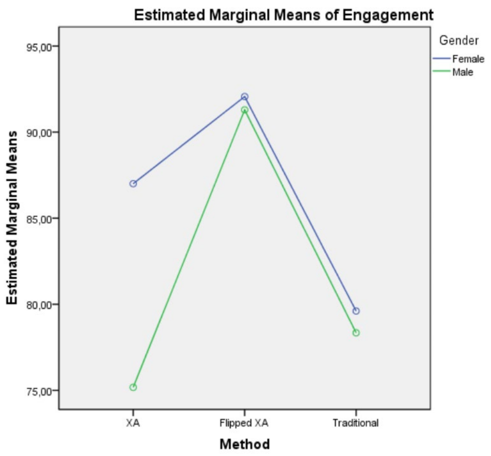


Fig. 5 Estimated marginal means of engagement scores

flipped XA and traditional in favor of flipped XA. Engagement scores show a significant difference between XA and flipped XA as well as between flipped XA and traditional in favor of flipped XA.

As seen in Fig. 5, the highest engagement is seen in the flipped XA group. While the engagement of female students in the XA group was higher than that of female students in the traditional group, the opposite was true for male students.

As seen in Fig. 6, regardless of gender, XA was the most effective method on achievement followed by flipped XA, and traditional was the least effective. It had already been predicted that the other two methods would be more effective compared to traditional. However, while it was thought that flipped XA could be more effective than XA, the opposite was in fact true and this is worth some discussion.

5 Discussion

This study compares the applications of the extreme apprenticeship method, flipped extreme apprenticeship, and traditional classroom, analyzing them at the university level in terms of their effects on academic achievement and engagement coupled with gender differences. The study found significant differences in terms of achievement and engagement in all three learning environments. The findings are discussed in light of the literature in the following sections.

5.1 Achievement of the students in terms of the XA, flipped XA and traditional methods

In this study, the XA method was the most effective on achievement. Learners were able to visibly improve their programming skills by completing numerous short exercises and then receiving feedback. Programmers who have a good experience in programming tend to build stronger program structures and in a shorter amount of time

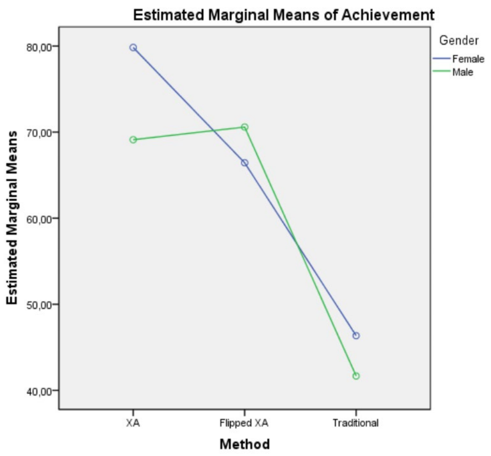


Fig. 6 Estimated marginal means of achievement scores

than those who have poor experiences (Adelson, 1981). In this respect, the method adopted to teach programming is essential. The learners involved in this study were not only more successful but also had a better learning experience. They received immediate feedback showing them where they had gone wrong, and then had the opportunity to come up with new solutions while being coached by the instructor. Thus, learners gained valuable experience through the XA process, a method that combines theory and practice with multiple programming tasks ranging from simple to complex (Leinonen et al., 2019). The fact that learners performed well in these environments has also been found in previous research studies, corroborating this study's findings (Keijonen et al., 2013; Pasini et al., 2016). Vihavainen et al. (2011b) compared the programming courses they previously taught using the traditional method to the courses, they taught using XA in subsequent periods. They found that the courses using XA had a higher pass rate. Other studies have previously been done on this subject (Keijonen et al., 2013; Pasini et al., 2016; Vihavainen et al., 2011b), but by going one step further and comparing the exam success scores of the control group and the experimental group using statistical analyses, this study obtained a higher effect size providing clearer evidence that XA is a more effective method.

When the XA and flipped class methods were used, achievement was greater than in the traditional environment. This finding is in line with research in the field comparing flipped and traditional learning environments for programming instruction (Amresh et al. 2013; Chen et al., 2014; Chis et al., 2018; Pattanaphanchai, 2019; Puarungroj, 2015; Souza & Rodrigues 2015). What is noteworthy here, however, is that achievement in the XA group was higher compared to the flipped XA environment. However, the average in the flipped XA environment was not much smaller than the average in the XA environment. Therefore, some adjustments can be made in the flipped XA environment to achieve higher achievement. Moreover, merely examining achievement may not be adequate in terms of shaping learning environments. Observations made during the learning process are just as important as final achievement in determining the impact of a methodology used in teaching programming skills (Vihavainen et al., 2013).

5.2 Student engagement in terms of the XA, flipped XA, and traditional methods

Because of the complexities of programming, learners can lose motivation when no good instructional method is available, and they may even drop out of a course entirely. Research shows that a significant number of students drop out of programming classes because they are struggling (McKinney & Denton, 2004). This is alarming in today's world, where computers are used in almost every aspect of life and programming skills are crucial. This study examined the rates of doing homework, attendance, and video views in order to reveal the students' engagement status in detail. The XA method adopted in this study significantly increased the engagement of learners in the course as well as their achievement. The flipped XA environment had the highest engagement rate of the three study groups. Similarly, in their study, Chis et al. (2018) found that students were more satisfied with the flipped learning environment and engaged more in lessons compared to traditional programming lessons. However, in the current study, while students were more involved, their gains were not as great as in the XA environment. Factors affecting engagement should be considered in this case.

In this study, video watching rates, homework, and attendance determined engagement. Of these three elements, only video is unique to the flipped environment. Videos are important for learners to get a sense of course topics before class and to achieve higher levels of learning by bringing problems they have to be solved to the classroom (Yıldız Durak, 2018). However, in spite of the high level of engagement in this study, the low achievement rate suggests that there may be issues with the students' individual learning processes. Previous studies have emphasized that learners may not adapt to the flipped environment, face problems focusing, and often do not efficiently manage their learning processes well (Vivek & Ramkumar, 2021). In this regard, the students may have watched videos tailored to the flipped environment out of necessity, but they may not have gained meaningful learning experiences as a result. Previous research has shown that videos can increase cognitive load (Homer et al., 2008). Above a certain level, increased cognitive load negatively affects learning performance (Sweller, 1994). The duration of the video, the incorporation of complex subjects such as algorithms and syntax, the creation of programming structures, and the inability to ask questions about problems being experienced at the time may have increased cognitive load and reduced attention and learning.

Studies done using flipped learning show that, in general, the learning environment is more effective in terms of engagement than the traditional environment. According to Puarungroj (2015), in flipped environments, learners can become more motivated and engaged in lessons more because of videos. Giannakos, Krogstie, and Chrisochoides (2014b) examined research results in the literature and concluded that flipped learning had a positive effect on student performance and engagement when designed well. In an introduction to programming course, Kuzminska et al. (2017) compared the flipped environment to the traditional method. According to their results, in the flipped environment, students' homework and course engagement were more positive than in the traditional environment. Since engagement is important for achievement when learning programming, the flipped XA environment is considered to offer an important infrastructure.

In future studies, learner engagement in the flipped XA environment should be closely monitored, and the difficulties encountered in the process should be identified and the reasons investigated. In this regard, the instructor's performance in the flipped environment is just as important as the learner's. According to Dimauro et al. (2019), the flipped learning environment necessitates a significant effort on the part of instructors in their transition from knowledge-transmitters to coaches. Instructors must track students both inside and outside of the classroom to check whether they are watching videos and preparing for lessons. In addition, learners should be given the necessary guidance regarding how to work in flipped environments and the path they will take. This, however, can be exhausting for the instructor. It takes time to prepare videos before class, add questions to videos, monitor students, and plan the necessary guidance. Moreover, some students may be completely unprepared for lessons (Triantafyllou & Timcenko, 2014), necessitating the re-teaching of basic course content. It is deemed critical that educational institutions provide necessary support to instructors in order to ensure that instructor performance does not deteriorate and engagement is increased through improved course processes.

Another consideration is that learners who are accustomed to the traditional learning environment may find the flipped learning environment challenging (Amresh et al., 2013; Loftsson et al., 2019). This could be another reason why students did not perform as well as expected in the flipped XA environment. There is a significant decrease in the desire to learn programming, particularly when the instructional design is unsuitable for individuals (Lahtinen et al., 2005). According to Gündüz and Akkoyunlu (2019), it is concerning that some university students were unable to get feedback while watching videos at home prepared for programming learning in the flipped learning environment; they became bored watching videos alone, and some were unable to adequately watch videos at home due to internet problems. Loftsson and Matthíasdóttir (2019) stated that the rate of students watching the given videos same with the content to make them come to the lesson preparedly was insufficient.. According to Giannakoset al. (2014b), students may be unreceptive to the structure of the flipped environment, and attendance may suffer as a result. Tyler and Abdrakhmanova (2016) found that the flipped environment was beneficial to student engagement and performance only on average. In this regard, the flipped XA environment would be more effective when learners are ready based on their individual characteristics and instructors have adequate resources and support.

It is also important for those who learn in a flipped environment to have control of their own learning and to possess self-directed learning skills because they will need to improve their programming skills on their own at home (Yıldız Durak, 2020). Tracking and improving self-directed learning and the learner control processes (Knowles, 1975), which include learners determining their needs and learning goals, employing appropriate learning strategies, and evaluating their learning processes, would be beneficial in improving learning performance in a flipped environment (Çakıroğlu & Öztürk, 2017; Yıldız Durak, 2018). While watching videos, students may become distracted by social media sites such as Facebook or use chatting apps (Puarungroj, 2015). Students' self-directed learning skills, as well as social media addiction, can be managed in this regard. Moreover, they should be encouraged to take note of questions that arise while watching videos and bring them to the classroom.

There are some advantages to digital media in the home learning process that should be taken advantage. For example, to improve students' programming skills in an individualized learning environment, the instructional environment can be enriched with interactive animations, visuals, and voice communication. (Korhonen & Malmi, 2000). Social networks and online discussion environments can also be incorporated into flipped XA environments (Kuzminska et al., 2017). At the same time, engagement and participation may be increased by adding different elements attractive to learners such as gamification (Ekici, 2021).

5.3 Achievement and engagement of students in terms of gender and the three instructional methods

There was no difference in achievement and engagement levels of the students in terms of gender. Girls are more willing than boys to solve problems in complex areas such as programming (Lee, 2020), and girls tend to take a more systematic approach to developing solutions in their programming processes, first constructing a logical structure before writing lines of code (Funke et al., 2015). However, since females find programming more complicated and challenging than males, this may stifle their engagement, motivation, and performance (Akinola, 2015; Balanskat & Engelhardt, 2014; Giannakos et al., 2014a). However, society in general is not supportive enough for girls to progress in fields such as engineering and science (Yu et al., 2007). Boys, on the other hand, are more interested in and confident in programming (Maguire et al., 2014; Rubio et al., 2015; Stoilescu & McDougall, 2011) due to their prior social and cultural experiences with programming as well as their increased exposure to technology (Kay, 2006; Loftsson et al., 2019). These two situations may have balanced the achievement and engagement in terms of both genders. Similarly, neither Tyler and Yessenbayeva (2018), who used videos and practice quizzes in a flipped environment, nor Pasini et al. (2017), who used the XA method in an introduction to programming course, found any differences in performance between genders. In addition, the researchers considered the different sample sizes of male and female students. The results of the study related to achievement and engagement in programming courses show similar effects due to gender difference as that reported in the literature (Choi, 2015; Pasini et al., 2017; Tyler & Yessenbayeva, 2018). Also, as stated in the method, the analysis indicated that the difference between the sample sizes would not affect the result (Hair et al., 2009; Van Huynh et al., 2018). However, in future studies, examining cases where sample numbers are close to each other will further contribute to the literature.

6 Conclusions

Our goal was to develop the most effective learning environment and experience that we can for novices to learn programming. The results of this study will be useful in designing programming instruction environments for novice learners. Owing to the XA environment in this study, learners were much more successful in programming and engaged at a higher level than in the traditional environment, with the help of gradually decreasing guidance ranging from very much to little, and a handful of small exercises that were not tedious. On the other hand, flipped XA was more effective than the traditional environment in terms of achievement, but it was mildly behind the XA environment. For students to be experts, they must assume the necessary responsibility in their own learning processes through participating in the course (Rämö et al., 2020). Students who are unfamiliar with the flipped environment, however, may become distracted or lose motivation. It is also essential to plan educational environments that can be adapted to suddenly changing situations, for example pandemics such as coronavirus (COVID-19). In such cases, as they may have less contact with the school environment, learners may need to manage their own learning processes more by navigating resources and online environments. Studies show that the flipped learning environment positively affected learners' participation and motivation during the coronavirus pandemic by keeping them more active (Ng et al., 2022). With flipped learning, negativities such as the dullness of the traditional environment and the loneliness caused by the online environment, such as dissatisfaction and reluctance to participate, can be eliminated (Priyaadharshini & Vinayaga Sundaram, 2018; Tang et al., 2020). In this study, the flipped XA and XA environment significantly increased engagement. What is more, the XA method may enable students to achieve a higher level of achievement as they transition from being a novice to a master in the learning area with the increase in the number of activities, increased feedback and more online and face-to-face classes, as occurred during the COVID-19 pandemic. For this reason, it is also essential to study such environments. In this regard, the focus should be on how instructors can provide the best support to learners. Instructors can create virtual environments in which students can interact in flipped environments. Specific studies on videos can be conducted in the future, such as comparing them to standard videos in terms of length and interaction, to ensure that videos in the flipped environment do not become monotonous. Studies can be carried out to investigate the effectiveness of these videos in improving self-directed learning skills. A particular limitation of this study is the fact that it only looked at gender in terms of individual differences. Further studies may be conducted with more students looking at various individual differences such as personality traits and learning preferences. Such research results may also provide insight into advanced programming instruction environments. While the study includes a representative population of a metropolitan city university, it is limited to one education faculty. Future studies may be conducted with a larger sample size, including more than one faculty of education.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10639-022-11055-y.

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declarations

Conflicts of interest/Competing interests The authors declare that they have no conflict of interest.

Ethics approval The data collection procedures were approved by the university's ethical committee and participants were recruited following ethical standards.

References

Adelson, B. (1981). Problem solving and the development of abstract categories in programming languages. *Memory & cognition*, 9(4), 422–433. https://doi.org/10.3758/BF03197568

Akinola, S. O. (2015). Computer programming skill and gender difference: An empirical study. *American Journal of Scientific and Industrial Research*, 7(1), 1–9. https://doi.org/10.5251/ajsir.2016.7.1.1.9

Alvarado, C., Cao, Y., & Minnes, M. (2017). Gender differences in students' behaviors in CS classes throughout the CS major. In *Proceedings of the 2017 acm sigcse technical symposium on computer science education* (pp. 27–32). SIGCSE. https://doi.org/10.1145/3017680.3017771

Amresh, A., Carberry, A. R., & Femiani, J. (2013, October). Evaluating the effectiveness of flipped classrooms for teaching CS1. In *2013 IEEE Frontiers in Education Conference (FIE)* (pp. 733–735). IEEE. https://scholar.google.com/scholar?hl=tr&as_sdt=0%2C5&q=Evaluating+the+effectiveness+of+flipped+classrooms+for+teaching+CS1&btnG=

Balanskat, A., & Engelhardt, K. (2014). *Computing our future: Computer programming and coding - priorities, school curricula and initiatives across Europe*. European Schoolnet: Brussels, Belgium. http://www.eun.org/documents/411753/817341/Computing+our+future_final_2015.pdf/d3780a64-1081-4488-8549-6033200e3c03

Barnette, J. J., & McLean, J. E. (2005). Type I error of four pairwise mean comparison procedures conducted as protected and unprotected tests. *Journal of Modern Applied Statistical Methods*, 4(2), 10. https://doi.org/10.22237/jmasm/1130803740

Bean, N., Weese, J., Feldhausen, R., & Bell, R. S. (2015). Starting from scratch: Developing a pre-service teacher training program in computational thinking. In *2015 IEEE Frontiers in Education Conference (FIE)* (pp. 1–8). IEEE. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7344237

Black, T. R. (2007). *Cognitive Apprenticeship and the computer programming student: How do students learn within this framework?*. The University of New Mexico. USA

Bravo, C., Marcelino, M. J., Gomes, A. J., Esteves, M., & Mendes, A. J. (2005). Integrating Educational Tools for Collaborative Computer Programming Learning. *The International Journal of Universal Computer Science*, 11(9), 1505–1517. http://jucs.org/jucs_11_9/integrating_educational_tools_for/jucs_11_9_1505_1517_cbravo.pdf

Bruhn, R. E., & Burton, P. J. (2003). An approach to teaching Java using computers. *ACM SIGCSE Bulletin*, 35(4), 94–99. https://doi.org/10.1145/960492.960537

Carbonaro, A. (2019). Good practices to influence engagement and learning outcomes on a traditional introductory programming course. *Interactive Learning Environments*, 27(7), 919–926. https://doi.org/10.1080/10494820.2018.1504307

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

Chen, Y., Wang, Y., & Chen, N. S. (2014). Is FLIP enough? Or should we use the FLIPPED model instead? *Computers & Education*, 79, 16–27. https://doi.org/10.1016/j.compedu.2014.07.004

Chis, A. E., Moldovan, A. N., Murphy, L., Pathak, P., & Muntean, C. H. (2018). Investigating flipped classroom and problem-based learning in a programming module for computing conversion course. *Journal of Educational Technology & Society*, 21(4), 232–247. https://doi.org/10.2307/26511551. https://www.jstor.org/stable/

Choi, K. S. (2015). A comparative analysis of different gender pair combinations in pair programming. *Behaviour & Information Technology*, 34(8), 825–837. https://doi.org/10.1080/0144929X.2014.937460

Çakıroğlu, Ü., & Öztürk, M. (2017). Flipped classroom with problem based activities: Exploring self-regulated learning in a programming language course. *Journal of Educational Technology & Society*, 20(1), 337–349. https://doi.org/10.2307/jeductechsoci.20.1.337. https://www.jstor.org/stable/

Del Fatto, V., Dodero, G., & Gennari, R. (2016). How measuring student performances allows for measuring blended extreme apprenticeship for learning Bash programming. *Computers in Human Behavior*, 55, 1231–1240. https://doi.org/10.1016/j.chb.2015.04.007

Dimauro, G., Gentile, E., Plantamura, P., & Scalera, M. (2019). Experimentation of Flipped Learning in Higher Education Academy. *Int. J. Infonomics*, 12, 1891–1898. https://doi.org/10.20533/iji.1742.4712.2019.0194

Ekici, M. (2021). A systematic review of the use of gamification in flipped learning. *Education and Information Technologies*, 26(3), 3327–3346. https://doi.org/10.1007/s10639-020-10394-y

Fraenkel, J. R., Wallen, E. N., & Hyun, H. H. (2012). *How to design and evaluate research in education* (8th ed.). New York, NY: McGraw-Hill

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, 32(6), 576–593. https://doi.org/10.1111/jcal.12155

Funke, A., Berges, M., Mühling, A., & Hubwieser, P. (2015). Gender differences in programming: Research results and teachers' perception, *In Proceedings of the 15th Koli Calling Conference on Computer Education Research*, ACM, Finland. https://doi.org/dl.acm.org/doi/abs/10.1145/2828959.2828982

Giannakos, M. N., Jaccheri, L., & Leftheriotis, I. (2014a). Happy girls engaging with technology: Assessing emotions and engagement related to programming activities. *Lecture Notes in Computer Science, 8523*, 398–409 Springer

Giannakos, M. N., Krogstie, J., & Chrisochoides, N. (2014b). Reviewing the flipped classroom research: reflections for computer science education. In *Proceedings of the computer science education research conference* (pp. 23–29). CSERC. https://doi.org/dl.acm.org/doi/abs/10.1145/2691352.2691354

Gündüz, A. Y., & Akkoyunlu, B. (2019). Student views on the use of flipped learning in higher education: A pilot study. *Education and Information Technologies*, 24(4), 2391–2401. https://doi.org/10.1007/s10639-019-09881-8

Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2009). Multivariate Data Analysis (7th Ed.). Pearson

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

Hautala, T., Romu, T., Rämö, J., & Vikberg, T. (2012, July). Extreme apprenticeship method in teaching university-level mathematics. In *Pre-proceedings of the 12th International Congress on Mathematical Education*. https://helda.helsinki.fi/handle/10138/303211

Homer, B. D., Plass, J. L., & Blake, L. (2008). The effects of video on cognitive load and social presence in multimedia-learning. *Computers in Human Behavior*, 24(3), 786–797. https://doi.org/10.1016/j.chb.2007.02.009

Hopcan, S., Polat, E., & Albayrak, E. (2022). Collaborative Behavior Patterns of Students in Programming Instruction. *Journal of Educational Computing Research*, 07356331211062260. https://doi.org/10.1177/07356331211062260

Hwang, W. Y., Shadiev, R., Wang, C. Y., & Huang, Z. H. (2012). A pilot study of cooperative programming learning behavior and its relationship with students' learning performance. *Computers & Education*, 58(4), 1267–1281. https://doi.org/10.1016/j.compedu.2011.12.009

Iqbal Malik, S. (2016). *Role of ADRI model in teaching and assessing novice programmers* (Doctoral dissertation, Deakin University). Australia. https://dro.deakin.edu.au/view/DU:30088862

Kafai, Y. B., & Burke, Q. (2013). The social turn in K-12 programming: moving from computational thinking to computational participation. In *Proceeding of the 44th ACM technical symposium on computer science education* (pp. 603–608)

Kay, R. (2006). Addressing gender differences in computer ability, attitudes and use: The laptop effect. *Journal of Educational Computing Research*, 34(2), 187–211. https://doi.org/10.2190/9BLQ-883Y-XQMA-FCAH

Keijonen, H., Kurhila, J., & Vihavainen, A. (2013). Carry-on effect in extreme apprenticeship. In *2013 IEEE Frontiers in Education Conference (FIE)* (pp. 1150–1155). IEEE. https://ieeexplore.ieee.org/abstract/document/6685011/?casa_token=9Q1XQ1Aw1HQAAAAA:9lLYodOx4kTdJDLFw9SEz5JmHxqS8Ulsdx5ReCIPVlr_uEFG4b_6FHlkrrXm1vDwiKXvC3UFJaY

King, A. (1993). From sage on the stage to guide on the side. *College Teaching*, 41(1), 30–35. https://doi.org/10.1080/87567555.1993.9926781

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2), 75–86. https://doi.org/10.1207/s15326985ep4102_1

Knowles, M. (1975). *Self-directed learning: A guide for learners and teachers*. New York, NY: Associated Press

Korhonen, A., & Malmi, L. (2000). Algorithm simulation with automatic assessment. In *Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education* (pp. 160–163). https://doi.org/dl.acm.org/doi/abs/10.1145/343048.343157

Koulouri, T., Lauria, S., & Macredie, R. D. (2014). Teaching introductory programming: A quantitative evaluation of different approaches. *ACM Transactions on Computing Education (TOCE)*, 14(4), 1–28. https://doi.org/10.1145/2662412

Kuzminska, O., Morze, N., & Smyrnova-Trybulska, E. (2017). Flipped learning model: Tools and experience of its implementation in higher education. *The New Aducational Review*, 49(3), https://doi.org/10.15804/tner.2017.49.3.15

Lahdenperä, J., Postareff, L., & Rämö, J. (2019). Supporting quality of learning in university mathematics: A comparison of two instructional designs. *International Journal of Research in Undergraduate Mathematics Education*, 5(1), 75–96. https://doi.org/10.1007/s40753-018-0080-y

Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin*, 37(3), 14–18. https://doi.org/10.1145/1151954.1067453

Lee, K. (2020). Effects of Teaching Method on the Self-efficacy in a University Programming Class. In *Proceedings of the 18th ROME - ITALY International Conference on Art History, Literature, Social Sciences and Education* (pp. 20–25). https://doi.org/10.17758/HEAIG7.H0220426

Lee, Y. J. (2011). Empowering teachers to create educational software: A constructivist approach utilizing Etoys, pair programming and cognitive apprenticeship. *Computers & Education*, 56(2), 527–538. https://doi.org/10.1016/j.compedu.2010.09.018

Leinonen, J., Ihantola, P., Leinonen, A., Nygren, H., Kurhila, J., Luukkainen, M., & Hellas, A. (2019). Admitting Students through an Open Online Course in Programming: A Multi-year Analysis of Study Success. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (pp. 279–287). https://doi.org/dl.acm.org/doi/abs/10.1145/3291279.3339417

Loftsson, H., & Matthíasdóttir, Á. (2019). In *2019 International Conference on Engineering, Technology and Education (TALE)* (pp. 1–6). IEEE. https://ieeexplore.ieee.org/abstract/document/9225985?casa_token=4tRunuksP7EAAAAA:dKt3Vh-Ub60GvzC3l0rxS-G0Fwik9DM5fJXLbe7YN4IAwFOu_hDZMLGUPWJvjvdZq8SK59eKQX4

Maguire, P., Maguire, R., Hyland, P., & Marshall, P. (2014). Enhancing collaborative learning using pair programming: Who benefits? *AISHE-J: The All Ireland Journal of Teaching and Learning in Higher Education*, 6(2), 141. http://ojs.aishe.org/index.php/aishe-j/article/view/141

Maher, M. L., Latulipe, C., Lipford, H., & Rorrer, A. (2015). Flipped classroom strategies for CS education. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 218–223)

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM*, 49(8), 90–95. https://doi.org/10.1145/1145287.1145293

McKinney, D., & Denton, L. F. (2004). Houston, we have a problem: there's a leak in the CS1 affective oxygen tank. *ACM SIGCSE Bulletin*, 36(1), 236–239. https://doi.org/10.1145/1028174.971386

Ng, D. T., Ng, E. H., & Chu, S. K. (2022). Engaging students in creative music making with musical instrument application in an online flipped classroom. *Education and information Technologies*, 27(1), 45–64. https://doi.org/10.1007/s10639-021-10568-2

Nikula, U., Gotel, O., & Kasurinen, J. (2011). A motivation guided holistic rehabilitation of the first programming course. *ACM Transactions on Computing Education (TOCE)*, 11(4), 1–38. https://doi.org/10.1145/2048931.2048935

Pallant, J. (2005). *SPSS survival manual: A step by step guide to data analysis using SPSS for Windows*. Australia: Allen & Unwin

Pasini, M., Solitro, U., Brondino, M., & Raccanello, D. (2016, September). The Challenge of Learning to Program: motivation and achievement emotions in an eXtreme Apprenticeship experience. In: Church, L. (ed.) *27th Annual Workshop of the Psychology of Programming Interest Group - PPIG 2016* (pp. 150–155). University of Cambridge, UK. https://iris.univr.it/retrieve/handle/11562/954662/59535/PPIG-final20.pdf

Pasini, M., Solitro, U., Brondino, M., & Raccanello, D. (2017). The role of the cognitive style in improving the learning to program. In: Church, L. (ed.) *27th Annual Workshop of the Psychology of Programming Interest Group, PPIG*, (pp. 150–155). https://www.ppig.org/files/2017-PPIG-28th-pasini.pdf

Pattanaphanchai, J. (2019). An Investigation of Students' Learning Achievement and Perception using Flipped Classroom in an Introductory Programming Course: A Case Study of Thailand Higher Education. *Journal of University Teaching and Learning Practice*, 16(5), 4. https://ro.uow.edu.au/jutlp/vol16/iss5/4

Plonka, L., Sharp, H., Van der Linden, J., & Dittrich, Y. (2015). Knowledge transfer in pair programming: An in-depth analysis. *International Journal of Human-Computer Studies*, 73, 66–78. https://doi.org/10.1016/j.ijhcs.2014.09.001

Polat, E., & Hopcan, S. (2019). Teachers' Acceptance of MIT App Inventor as an Educational Mobile Application Development Tool. *Kastamonu Educational Journal*, 27(6), 2459–2466. https://doi.org/10.24106/kefdergi.3300

Priyaadharshini, M., & Vinayaga Sundaram, B. (2018). Evaluation of higher-order thinking skills using learning style in an undergraduate engineering in flipped classroom. *Computer Applications in Engineering Education*, 26(6), 2237–2254. https://doi.org/10.1002/cae.22035

Puarungroj, W. (2015). Inverting a computer programming class with the flipped classroom. In *Proceedings of the the Twelfth International Conference on eLearning for Knowledge-Based Society*, (pp. 11–12). http://www.elearningap.com/eLAP2015/Proceedings/02_40_Inverting.pdf

Rämö, J., Oinonen, L., & Vikberg, T. (2015). Extreme Apprenticeship–Emphasising conceptual understanding in undergraduate mathematics. In *CERME9 Proceedings of the Ninth Congress of the European Society for Research in Mathematics Education*. Charles University in Prague, Faculty of Education and ERME. https://helda.helsinki.fi/bitstream/handle/10138/303099/publication4_Ramo_Johanna.pdf?sequence=1

Rämö, J., Lahdenperä, J., & Häsä, J. (2020). The extreme apprenticeship method. *PRIMUS*, 1–15. https://doi.org/10.1080/10511970.2020.1818332

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Rodríguez, F. J., Price, K. M., & Boyer, K. E. (2017). Exploring the pair programming process: Characteristics of effective collaboration. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, (pp. 507–512). https://doi.org/10.1145/3017680.3017748

Rubio, M. A., Romero-Zaliz, R., Mañoso, C., & Angel, P. (2015). Closing the gender gap in an introductory programming course. *Computers & Education*, 82, 409–420. https://doi.org/10.1016/j.compedu.2014.12.003

Sarawagi, N. (2013). Flipping an introductory programming course: Yes you can!. *Journal of Computing Sciences in Colleges*, 28(6), 186–188. https://doi.org/dl.acm.org/doi/abs/10.5555/2460156.2460190

Sinha, T., & Kapur, M. (2021). Robust effects of the efficacy of explicit failure-driven scaffolding in problem-solving prior to instruction: A replication and extension. *Learning and Instruction*, 75, 101488. https://doi.org/10.1016/j.learninstruc.2021.101488

Solitro, U., Zorzi, M., Pasini, M., & Brondino, M. (2016). A "light" application of blended extreme apprenticeship in teaching programming to students of mathematics. *Methodologies and Intelligent Systems for Technology Enhanced Learning* (pp. 73–80). Cham: Springer. https://link.springer.com/chapter/https://doi.org/10.1007/978-3-319-40165-2_8

Souza, M. J. D., & Rodrigues, P. (2015). Investigating the effectiveness of the flipped classroom in an introductory programming course. *The New Educational Review*, 40(2), 129–139. https://doi.org/10.15804/tner.2015.40.2.11

Spohrer, J. C., & Soloway, E. (1986). Novice mistakes: Are the folk wisdoms correct? *Communications of the ACM*, 29(7), 624–632. https://doi.org/10.1145/6138.6145

Stoilescu, D., & McDougall, D. (2011). Gender digital divide and challenges in undergraduate computer science programs. *Canadian Journal of Education*, 34(1), 308–333. https://doi.org/10.2307/canajeducrevucan.34.1.308. https://www.jstor.org/stable/

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. *Learning and Instruction*, 4(4), 295–312. https://doi.org/10.1016/0959-4752(94)90003-5

Tang, T., Abuhmaid, A. M., Olaimat, M., Oudat, D. M., Aldhaeebi, M., & Bamanger, E. (2020). Efficiency of flipped classroom with online-based teaching under COVID-19. *Interactive Learning Environments*, 1–12. https://doi.org/10.1080/10494820.2020.1817761

Taşpolat, A., Özdamli, F., & Soykan, E. (2021). Programming language training with the flipped classroom model. *Sage Open*, 11(2), https://doi.org/10.1177/21582440211021403

Triantafyllou, E., & Timcenko, O. (2014). Introducing a flipped classroom for a statistics course: A case study. In *2014 25th EAEEIE Annual Conference (EAEEIE)* (pp. 5–8). IEEE. https://ieeexplore.ieee.org/abstract/document/6879373?casa_token=7HuUjHvWocEAAAAA:AL6qSVfgRsaH5Sh-p93k2JSOdrKUFc465×0wh-h5L2Uui6jPNFKHZuhe_20_YZ_2Hg68gyI-bUk

Turnbull, D., Chugh, R., & Luck, J. (2021). The Use of Case Study Design in Learning Management System Research: A Label of Convenience? *International Journal of Qualitative Methods*, 20, 160940692110041. https://doi.org/10.1177/16094069211004148

Tyler, B., & Abdrakhmanova, M. (2016). Flipping the CS1 and CS2 classrooms in Central Asia. In *2016 IEEE Frontiers in Education Conference (FIE)* (pp. 1–5). IEEE. https://ieeexplore.ieee.org/abstract/document/7757739

Tyler, B., & Yessenbayeva, A. (2018). A Comparison of Flipped Programming Classroom Models–Results by Gender and Major. In *2018 IEEE Frontiers in Education Conference (FIE)* (pp. 1–5). IEEE. https://ieeexplore.ieee.org/abstract/document/8658809?casa_token=70MX2AKQDJgAAAAA:8vCSFapRIvDvoASgmVgn5oDlqr-fPW7W85ZQlCI3cIbuk7GTnAIepZH8yZ3rgogLnVxTlx0w6BM

Van Huynh, S., Tran-Chi, V. L., & Nguyen, T. T. (2018). Vietnamese teachers' perceptions of social-emotional learning education in primary schools. *European Journal of Contemporary Education*, 7(4), 874–881. https://doi.org/10.13187/ejced.2018.4.874

Vihavainen, A., Luukkainen, M., & Kurhila, J. (2012). Multi-faceted support for MOOC in programming. In *Proceedings of the 13th annual conference on Information technology education* (pp. 171–176). https://doi.org/10.1145/2380552.2380603

Vihavainen, A., Paksula, M., & Luukkainen, M. (2011a). Extreme apprenticeship method in teaching programming for beginners. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 93–98). ACM. https://doi.org/10.1145/1953163.1953196

Vihavainen, A., Paksula, M., Luukkainen, M., & Kurhila, J. (2011b). Extreme apprenticeship method: key practices and upward scalability. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (pp. 273–277). https://doi.org/10.1145/1999747.1999824

Vihavainen, A., Vikberg, T., Luukkainen, M., & Pärtel, M. (2013). Scaffolding students' learning using test my code. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education* (pp. 117–122). https://doi.org/10.1145/2462476.2462501

Vivek, C. M., & Ramkumar, P. (2021). Evaluation of course outcome attainment of engineering course with traditional, blended and flipped classroom approaches. *Education and Information Technologies*, 26(2), 2225–2231. https://doi.org/10.1007/s10639-020-10353-7

Wiedenbeck, S. (2005). Factors affecting the success of non-majors in learning to program. In *Proceedings of the first international workshop on Computing education research* (pp. 13–24). https://doi.org/10.1145/1089786.1089788

Williams, L., McDowell, C., Nagappan, N., Fernald, J., & Werner, L. (2003). Building pair programming knowledge through a family of experiments. In *2003 International Symposium on Empirical Software Engineering*. (pp. 143–152). IEEE. https://ieeexplore.ieee.org/abstract/document/1237973?casa_token=Zn6VsXVPVSkAAAAA:p19AUQgG6puLTaOlLJ63GRFyMNs-qZNp7-so-L5Hsa8SNhYhc5YGpL9yTPjuxa-1MMCF54yZvIs

Witherspoon, E. B., Schunn, C. D., Higashi, R. M., & Baehr, E. C. (2016). Gender, interest, and prior experience shape opportunities to learn programming in robotics competitions. *International Journal of STEM Education*, 3(1), 1–12. https://doi.org/10.1186/s40594-016-0052-1

Yıldız Durak, H. (2018). Flipped learning readiness in teaching programming in middle schools: Modelling its relation to various variables. *Journal of Computer Assisted Learning*, 34(6), 939–959. https://doi.org/10.1111/jcal.12302

Yıldız Durak, H. (2020). Modeling different variables in learning basic concepts of programming in flipped classrooms. *Journal of Educational Computing Research*, 58(1), 160–199. https://doi.org/10.1177/0735633119827956

Yu, E., Iskander, M., Kapila, V., & Kriftcher, N. (2007). Promoting engineering careers using modern sensors in high school science labs. In I. Magued (Ed.), *Innovations in e-learning, instruction technology, assessment, and engineering education* (pp. 229–235). Dordrecht: Springer

Zha, S., Jin, Y., Moore, P., & Gaston, J. (2020). Hopscotch into coding: introducing pre-service teachers computational thinking. *TechTrends*, 64(1), 17–28. https://doi.org/10.1007/s11528-019-00423-0