Computers and Education: Artificial Intelligence 4 (2023) 100147

Contents lists available at ScienceDirect

Computers and Education: Artificial Intelligence

journal homepage: www.sciencedirect.com/journal/computers-and-education-artificial-intelligence

The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation

Ramazan Yilmaz *, Fatma Gizem Karaoglan Yilmaz

Faculty of Science, Department of Computer Technology & Information Systems, Bartin University, Bartin, Turkey

ARTICLE INFO

Keywords:
Artificial intelligence
ChatGPT
Generative pretrained transformer
Programming education
Computational thinking

ABSTRACT

ChatGPT (generative pre-trained transformer) is one of the artificial intelligence (AI) technologies that have started to be used in programming education. However, the effect of using ChatGPT in programming education on learning processes and outcomes is not yet known. This study investigated the effect of programming education using the ChatGPT on students' computational thinking skills, programming self-efficacy, and motivation toward the lesson. The research was conducted on 45 undergraduate students who took a university-level programming course. The research was carried out according to the experimental design with the pretest-posttest control group. Students were randomly divided into experimental (n = 21) and control (n = 24) groups. While the experimental group students benefited from the ChatGPT during the weekly programming practices, the control group students did not use this tool. Research data were obtained through the computational thinking scale, computer programming self-efficacy scale, and learning motivation in computer programming courses scale. Research findings revealed that the experimental group students' computational thinking skills, programming self-efficacy, and motivation for the lesson were significantly higher than the control group students. In line with this result, it can be said that it may be useful to benefit from AI technologies such as ChatGPT in programming trainings. The research findings, it was emphasized how the most effective use of AI support in the lessons could be made, and various suggestions were made for researchers and educators in this regard.

1. Introduction

Computer programming is a necessary skill for many lines of business in today's modern economy. Having computer programming skills can give individuals the ability to create and build new technologies that can drive innovation and economic growth (Eteng et al., 2022; González-Pérez & Ramírez-Montoya, 2022; James, 2021). For this reason, employers today attach importance to employing individuals with computer programming skills. This applies to the technology industry and increasingly digitalized fields such as finance, health, transportation, and education. Educational institutions are trying to adapt to the needs that arise due to this change and change in today's business world. As a result, programming education is given on a wide scale, from early-age programming to adult education (Alam, 2022; Strawhacker & Bers, 2019).

Computer programming is the backbone of the internet and digital world, becoming increasingly important daily. Therefore, having strong programming skills can enable individuals to navigate and understand the digital environment more effectively. Thus, individuals can understand how these technologies work and how they can be used and manipulated. Computer programming is important for problem-solving and critical thinking (Mathew et al., 2019; Wang et al., 2017). Computer programming provides a clear and structured way to express ideas and solve problems that can be applied in many other fields. Even if one is not a software developer, being able to write code to solve problems can help individuals in many areas of life. Programming education is key to creativity and innovation (Liu et al., 2022a; Su et al., 2022). With computer programming skills, individuals can create new technologies and digital tools to drive innovation and economic growth.

Various teaching approaches are used to provide effective programming education to learners. Instructional approaches such as hands-on coding, project-based learning, pair programming, problem-based learning, and game-based learning are among the current approaches used in programming education in recent years

* Corresponding author.
E-mail addresses: ramazanyilmaz067@gmail.com (R. Yilmaz), gkaraoglanyilmaz@gmail.com (F.G. Karaoglan Yilmaz).

https://doi.org/10.1016/j.caeai.2023.100147
Received 5 April 2023; Received in revised form 7 June 2023; Accepted 7 June 2023
Available online 8 June 2023

(López-Pimentel et al., 2021; Malik et al., 2020; Sullivan & Strawhacker, 2021; Wei et al., 2021). The basis of these approaches lies in the fact that students learn to program cooperatively and in a fun way. However, it is stated in the literature that these educational approaches also have some disadvantages. One of these concerns is the challenges of working with others. Some students may find it difficult to work with others, especially when working on group projects, making it challenging to complete the assignment and learn effectively. In another dimension, which is seen as a disadvantage of collaborative learning approaches, the fact that the active student in the group assumes the leadership of the team is related to the fact that the other students contribute less to the process in the passive state (Yilmaz et al., 2020).

For this reason, it is essential that each student actively participates in the programming learning process individually and completes programming tasks. For this reason, the hands-on coding approach is one of the approaches that can be used in teaching programming to adult students. Hands-on coding is an effective method as it allows students to apply what they have learned immediately and helps them better understand and retain the material (Handur et al., 2016). Students may encounter problems in the learning process of programming, such as difficulties in understanding abstract concepts, debugging and troubleshooting, understanding the logic, mathematical concepts and application of programming, and keeping up with the pace of the class. In the hands-on coding process, external support providers may be needed to help the student overcome these problems. AI could provide a solution to the aforementioned problems. From this point of view, in this study, it is aimed to investigate the effect of AI-assisted programming education on students' programming skills and outcomes (computational thinking skills, programming self-efficacy, and motivation toward the programming lesson).

1.1. Literature review

In the teaching of programming using AI-based tools and environments, the student can ask the problem with the AI tool and can get instant feedback and solve the problem. Thus, the student can receive a personalized education suitable for his/her own learning pace (Yilmaz & Karaoglan Yilmaz, 2022a; 2022b). AI-powered tools can help students code by providing suggestions, error detection, and automatic code generation. This can help students write more efficient and accurate code and reduce the time and effort required to complete programming assignments. AI-powered tools and environments can increase student engagement and motivation by interacting with students and providing them with personalized support and feedback as they learn to program (Karaoglan Yilmaz & Yilmaz, 2022a; 2022b). When the literature is examined, it is seen that students cannot develop their computational thinking skills, have low self-efficacy in programming, and decrease in their motivation towards the lesson, which is among the main problems encountered in programming education (Fagerlund et al., 2021; Figueiredo & García-Peñalvo, 2020a; Liu et al., 2022b; Lye & Koh, 2014; Tikva & Tambouris, 2021; Tsai, 2019). Considering the advantages mentioned above of artificial intelligence-supported tools and environments, it is thought that it can be effective in improving students' computational thinking skills and increasing their programming self-efficacy and motivation toward the lesson. However, when the literature is examined, it is seen that the number and variety of research examining the effectiveness of AI support in programming education is low, and the application of AI in programming education is still in its early stages. It is seen that there is a need for new research results in this area.

Although there are no studies in the literature examining the effects of using ChatGPT in programming education on learning processes and outcomes, it is seen that various studies have been conducted recently on its use for educational purposes in general. It is seen that the majority of these studies are review articles that include evaluations on how ChatGPT can be used for educational purposes (Kasneci et al., 2023; Lo, 2023; Tlili et al., 2023). However, the effects of ChatGPT-supported education on students' learning processes and outcomes seem to be a gap in the literature that needs to be examined. Although there are no studies in the literature examining the effects of using ChatGPT in programming education on learning processes and outcomes, it is seen that various studies have been conducted recently on its use for educational purposes in general. It is seen that the majority of these studies are review articles that include evaluations on how ChatGPT can be used for educational purposes (Kasneci et al., 2023; Lo, 2023; Tlili et al., 2023). However, the effects of ChatGPT-supported education on students' learning processes and outcomes seem to be a gap in the literature that needs to be examined.

In this research, programming training was given using the ChatGPT tool, a large language model developed by OpenAI. The ChatGPT is based on the GPT (Generative Pre-training Transformer) architecture and is trained on a large text dataset from the internet. The ChatGPT is designed to generate human-like text and can be fine-tuned for specific tasks such as answering questions, language translation, and summarizing text (OpenAI, 2023). OpenAI is a research company founded in 2015 and its aim is to develop artificial intelligence technology. ChatGPT is part of OpenAI's GPT (Generative Pre-trained Transformer) series. Five different versions of ChatGPT have been released so far. The first version of ChatGPT, GPT-1, was released in 2018. This model is trained on a large language dataset (like Wikipedia) and has about 117 million parameters. Although the GPT-1 was considered a fairly large model for that period, it performed poorly when compared to later models. GPT-2, the second version of ChatGPT, was released in 2019. This model is a language model with approximately 1.5 billion parameters and is trained on a much larger dataset than previous models. GPT-2 has made significant progress in producing more natural and consistent language. However, some features of the model, such as its mass production capability, have been published on a limited basis as it raises abuse concerns. The third version of ChatGPT, GPT-3, was released in 2020. This model is trained on an even larger dataset compared to previous versions and has approximately 175 billion parameters. The GPT-3 can produce human-like natural texts and can be used for many different tasks. After the GPT-3 model, an intermediate model GPT-3.5 was published. Today, the GPT-4 version has started to be used. Thanks to its broad general knowledge and problem-solving capabilities, the GPT-4 is able to solve difficult problems with greater accuracy. It is stated that GPT-4 is more creative and collaborative compared to previous versions (OpenAI, 2023). The ChatGPT language model was used in the research and is based on the GPT-3.5 architecture (GPT-3.5 is a variation of GPT-3).

In this study, we control for the students' pretest scores, and look into whether there is a significant difference between the students of the experimental and control group on the following aspects:

RQ1. Computational thinking scale (creativity, algorithmic thinking, cooperativity, critical thinking, problem-solving) scores.

RQ2. Programming self-efficacy scale (simple programming tasks, complex programming tasks) scores.

RQ3. Motivation scale (individual attitude and expectation, challenging goals, clear direction, reward and recognition, punishment, social pressure and competition) scores.

2. Method

2.1. Research model and participants

This research was carried out according to the experimental design with the pretest-posttest control group. Experiment design with a pretest-posttest control group involved randomly assigning experiment participants as the experimental group and control group. Both groups were pretested for research purposes. The pretests of this research were

the computational thinking scale, the computer programming self-efficacy scale, and the learning motivation in computer programming courses scale. Then the experimental process begun. The experimental group received the intervention, while the control group did not. The intervention made to the experimental group within the scope of this research is the individual use of the ChatGPT tool in the laboratory assignments in computer programming education. The experimental process continued for five weeks. The reason why the experimental research was continued for five weeks is that the students in the experimental group absorbed the intervention well. Thus, students' thoughts about the intervention may have become more evident. At the end of the experimental process, both groups were subjected to a posttest to measure scores after the intervention. The posttests of this research were the computational thinking scale, the computer programming self-efficacy scale, and the learning motivation in computer programming courses scale. The difference between the pretest and posttest scores in the experimental group was compared with the difference in the control group to determine whether the intervention had a significant effect. The methodological representation of the research is given in Fig. 1.

The research was carried out on undergraduate students studying computer science at Bartin University in Turkey. Sixty-six students who took the object-oriented programming course participated in the research voluntarily. However, some of the students did not attend the lesson during the process, and some students did not answer the pretest and posttests. Therefore, the research was conducted on 45 students who attended the course and answered the pretest and posttest. There were 21 students in the experimental group and 24 students in the control group. Eleven of the students participating in the study were female, and 34 of them were male. The ages of the students participating in the research vary between 18 and 24. Within the scope of the research, students made object-oriented programming applications using the Java programming language. The students participating in the research did not receive any training in Java and object-oriented programming before. Consequently, it can be said that the students' prior knowledge and skills related to the subject are similar. There are several reasons for conducting the research in the programming course. Firstly, it was wondered how the advantages of ChatGPT in writing code would affect students' programming skills. For this reason, it was decided to conduct the research on students taking programming education course. Another reason for including these students in the study was that one of the researchers taught object-oriented programming to these students. Thus, it was aimed to prevent validity and reliability problems that may arise from instructor differences.

In this study, the following was done to ensure its validity and reliability. Participants were randomly assigned to either experimental or control groups to reduce bias and increase the internal validity of the study. A sufficiently large study group (45 students) was reached to increase the power of the study and reduce the probability of type 2 errors. Experimental group students were allowed to use the ChatGPT tool during the programming education applications. The control group students did not use the ChatGPT tool. Apart from this, the instructor of both groups, teaching methods, laboratory assignments, etc., are the same. In other words, apart from the intervention tool, there was no difference between the experimental and control groups that could affect the experimental process. All procedures performed in these studies were in accordance with the APA ethical guidelines, the ethical standards of the institutional research committee, and the 1964 Helsinki Declaration and its later amendments. Bartın University Scientific Research and Publication Ethics Guidelines were complied with for this study. In this framework, before starting the experimental process, the students in the experimental and control groups were informed about the aims and process of the research. It was explained what the students would do during the experimental process. In particular, it was explained why the experimental group students should use ChatGPT while doing their homework, and the control group should not use it. After the students were informed about the experiment, their consent was obtained for their participation in the experiment. The consent form

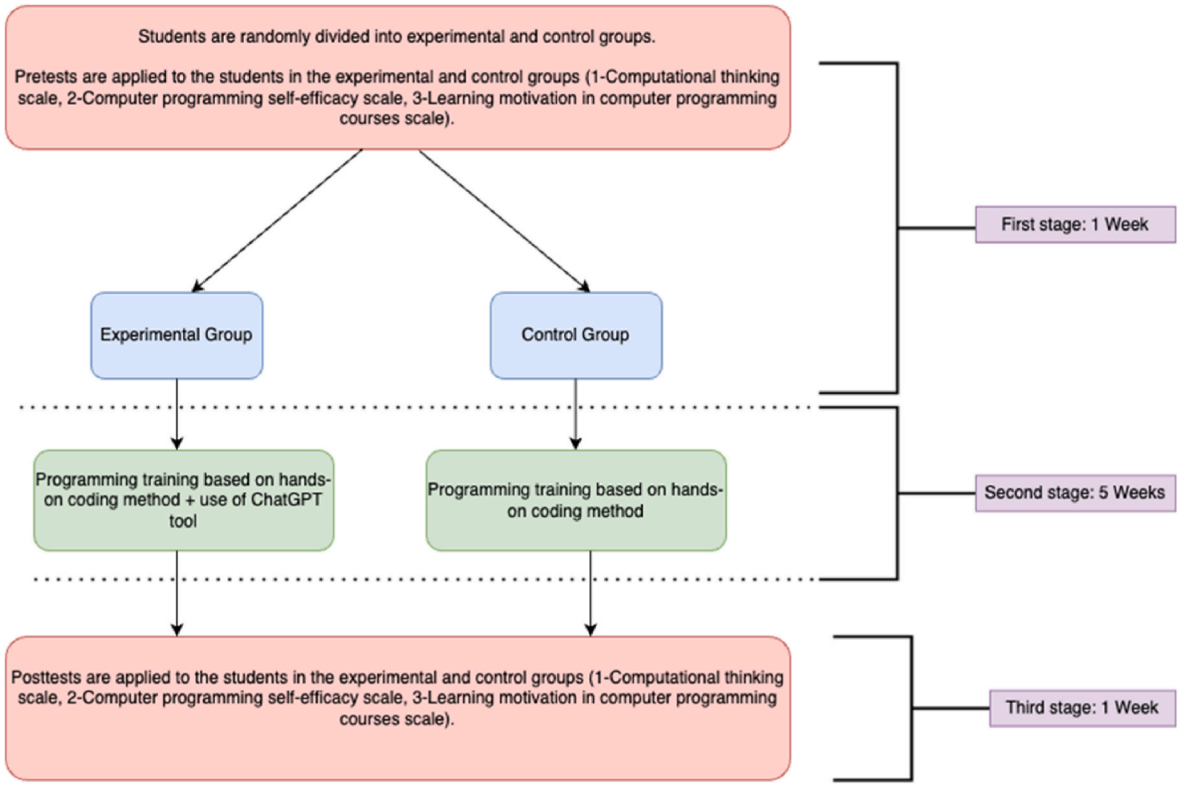

Fig. 1. Research process.

was written in plain language and the purpose of the study, the risks and benefits of participation, and the right to withdraw from the study at any time were explained to the students. The researchers also explained in the consent form that they would protect the confidentiality of students' personal information and research data.

2.2. Data collection tools

The scales used as pretest and posttest in the research are the computational thinking scale, the computer programming self-efficacy scale, and the learning motivation in computer programming courses scale. The explanations regarding the data collection tools used in the research are given below.

2.2.1. Computational thinking scale

Computational thinking scale was used to compare the computational thinking skills of the students in the experimental and control groups. The computational thinking scale was developed by Korkmaz et al. (2017). The scale consists of 29 items and five sub-factors: creativity, algorithmic thinking, cooperativity, critical thinking, and problem-solving. The scale uses a five-point Likert-type rating structure, with higher scores indicating greater development of computational thinking skills. In this study, the reliability of the scale was recalculated using Cronbach alpha reliability values, which were found to be 0.85 for creativity, 0.88 for algorithmic thinking, 0.87 for cooperativity, 0.73 for critical thinking, and 0.75 for problem-solving. The overall reliability value for the entire scale was calculated to be 0.84. The scale is given in Appendix-1.

2.2.2. Computer programming self-efficacy scale

The computer programming self-efficacy scale was adapted into Turkish by Altun and Mazman (2012) from the one developed by Ramalingam and Wiedenbeck (1998). The scale employs a seven-point Likert-type rating structure and comprises nine items categorized under two sub-factors: simple programming tasks and complex programming tasks. A higher score on the scale indicates advanced computer programming self-efficacy in students. The researchers recalculated the scale's reliability by examining Cronbach's alpha reliability values. The reliability values were as follows: 0.89 for simple programming tasks, 0.92 for complex programming tasks, and 0.88 for the entire scale.

2.2.3. Learning motivation in computer programming courses scale

The learning motivation in computer programming courses scale developed by Law et al. (2010) was adapted into Turkish by Avci and Ersoy (2018). The scale has a six-point Likert-type rating structure. The scale consists of 19 items and six sub-factors. These sub-factors are individual attitude and expectation, challenging goals, clear direction, reward and recognition, punishment, social pressure, and competition. A high score on the scale indicates that students have advanced learning motivation in computer programming courses. Within the scope of this research, the reliability of the scale was recalculated. For this, Cronbach alpha reliability values were examined. As a result of the analysis, the reliability values of the scale were determined as follows; individual attitude and expectation are 0.81, challenging goals 0.84, clear direction 0.73, reward and recognition 0.75, punishment 0.72, social pressure and competition 0.81. The reliability value calculated for the whole scale was found to be 0.85.

2.3. Research environment and procedure

The lesson was taught using a flipped classroom and a hands-on coding approach. Due to the flipped classroom approach, students prepare and come to the theoretical parts of the course before they come to the face-to-face class in the computer laboratory. In the face-to-face lesson in the computer laboratory, applications related to the topics of the week are made by using the hands-on coding approach. An account has been opened for the object-oriented programming course on the Moodle learning management system by the researchers. Then, materials related to weekly course topics were added to the learning management system. Lecture videos, presentations, e-books, and infographics were prepared for each week's topic by the researchers. Students study these course materials and come to the face-to-face class in the computer lab. After the instructor explains the week's subject in the computer laboratory course, the instructor makes the application homework about that subject accessible to the students through the learning management system. Students tried to do their application homework by using the explanations and course materials made by the instructor.

The face-to-face class in the computer lab lasted 2 h each week. Students completed the application homework during the course and send it to the instructor through the learning management system. With the hands-on coding approach in the course in the computer laboratory, the instructor first explained what will be done within the scope of the application and then asked the students to complete the application individually. What has been explained so far is applied similarly to the experimental and control group students. Unlike the face-to-face lesson in the computer laboratory, the students in the experimental group were allowed to use the ChatGPT tool while doing their application homework. At the beginning of the experimental process, the researchers explained to the experimental group students what the ChatGPT tool is, how it is used, and how they can benefit from it in the computer programming process. Experimental group students benefited from this tool while doing their weekly laboratory practices.

The ChatGPT tool can give correct answers for the desired simple coding applications. For example, "Can you write a program that calculates the average of two numbers in the java programming language?" It can give output as in Fig. 2. The code block shown in Fig. 2 is given in Appendix 2.

Since it was possible to do simple computer programming tasks with the ChatGPT, as in Fig. 2, the researchers asked the students to do complex and gradual laboratory practice assignments so that no direct answer could be found with the ChatGPT in weekly applications. An example of laboratory practice assignments is as follows.

Answer the following object-oriented programming language question using the java language.

Let's have a superclass named Product. Have information such as product name and price in our upper class. Let the Technology, Textile and Food classes inherit this superclass. In the textile class, let it have features such as product size and type. Have features such as brand and model in the technology class. Let the food class have type and weight information. The constructor method of each class will pass these imported properties to its properties in the class. Let the parent class have a VAT() method and subclasses will override this method. When the relevant information is sent to these objects from the main class, write the application that first prints the name of the class and then prints the price (including VAT) and class properties.

The ChatGPT's answer to the object-oriented programming question requested above is as in Fig. 3. The code block shown in Fig. 3 is given in Appendix 2.

It is seen that the ChatGPT correctly answers the complex object-oriented programming question above. For this reason, to prevent students from taking ready-made answers using the ChatGPT and using them in their practice assignments, weekly practice assignments were converted into Unified Modeling Language (UML) diagram structure, and what was required in the UML diagram in the laboratory environment was explained to the experimental and control group students. An example of the weekly laboratory assignments given to the experimental and control group students (related to the problem in Fig. 3) is shown in Fig. 4.

As seen in Fig. 4, weekly laboratory assignments are shown to the student as a UML diagram by the teacher. The teacher explains this

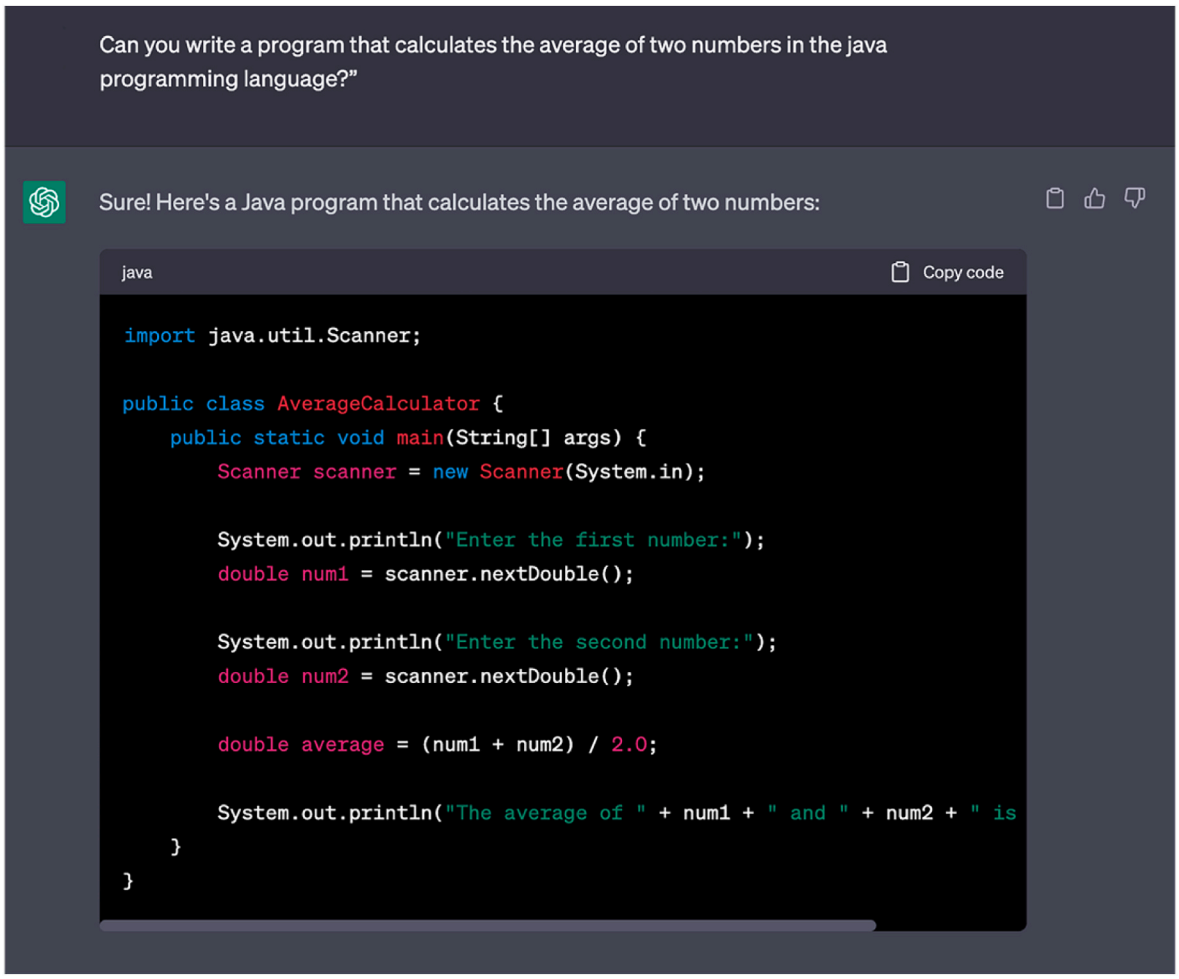

Fig. 2. ChatGPT's response to the desired simple programming application.

diagram to the students verbally, what the diagram means and what they will do according to the diagram. When Fig. 4 is examined, objects, variables, and methods are shown in the UML diagram. The student was accustomed to making an object-oriented programming application that will enable the desired output to be obtained. Experimental group students can use the ChatGPT to set up the structure shown in Fig. 4. Experimental group students were directed to this tool for students to benefit from the ChatGPT while solving the problem. Since ChatGPT is a text-based software, it is not yet capable of image processing. Therefore, to get the answer to the student's question from the ChatGPT, it is necessary to know what to ask and to think algorithmically. This is already for research purposes. In other words, to use ChatGPT, the student will have to develop their thinking skills. Whether this is effective or not was investigated within the scope of the study.

2.4. Data analysis

This study, which was carried out according to the pretest-posttest experimental design with the control group, was aimed to compare the scores of the experimental and control group students obtained from the scales before and after the experiment. First of all, the normality test was conducted with the Kolmogorov-Smirnov test to determine whether the scores obtained by the students from the scales showed a normal distribution. As a result of the analysis, it was determined that the data showed normal distribution. Then, the pretest scores of the students were controlled, and it was examined whether there was a significant difference between the posttest scores. ANCOVA test was performed for this review. ANCOVA test was used to determine the relationship between the dependent and independent variables and tries to explain this relationship. In other words, the ANCOVA test determined how the dependent variable changes based on the independent variables. Thus, it can be seen more clearly how the dependent variable changed based on the independent variables. Computational thinking scale, computer programming self-efficacy scale, learning motivation in computer programming courses scale pretest scores of the experimental and control group students were controlled, and their posttest scores were compared. ANCOVA test was used for this.

3. Findings

3.1. Findings on computational thinking skills

Table 1 provides the descriptive statistics for the computational thinking skills of the study groups.

Table 1 shows that the pretest scores of the students' computational thinking skills are similar to one another. To investigate the first research question and its sub-questions, a covariance analysis was carried out. In this case, the assumptions for the computational thinking skills scale were tested and found to be met. The ANCOVA analysis results are presented in Table 2.

Meanwhile, Table 2 indicates that the posttest scores for the experimental group's computational thinking skills (M = 126.73, SD = 8.34) were significantly higher than those of the control group (M = 112.61, SD = 15.32), with a medium effect size of [$F_{(1,42)}$: 18.760, p = .000, Cohen's f = 0.309] (Cohen, 1992).

The creativity posttest scores for the experimental group (M = 37.14,

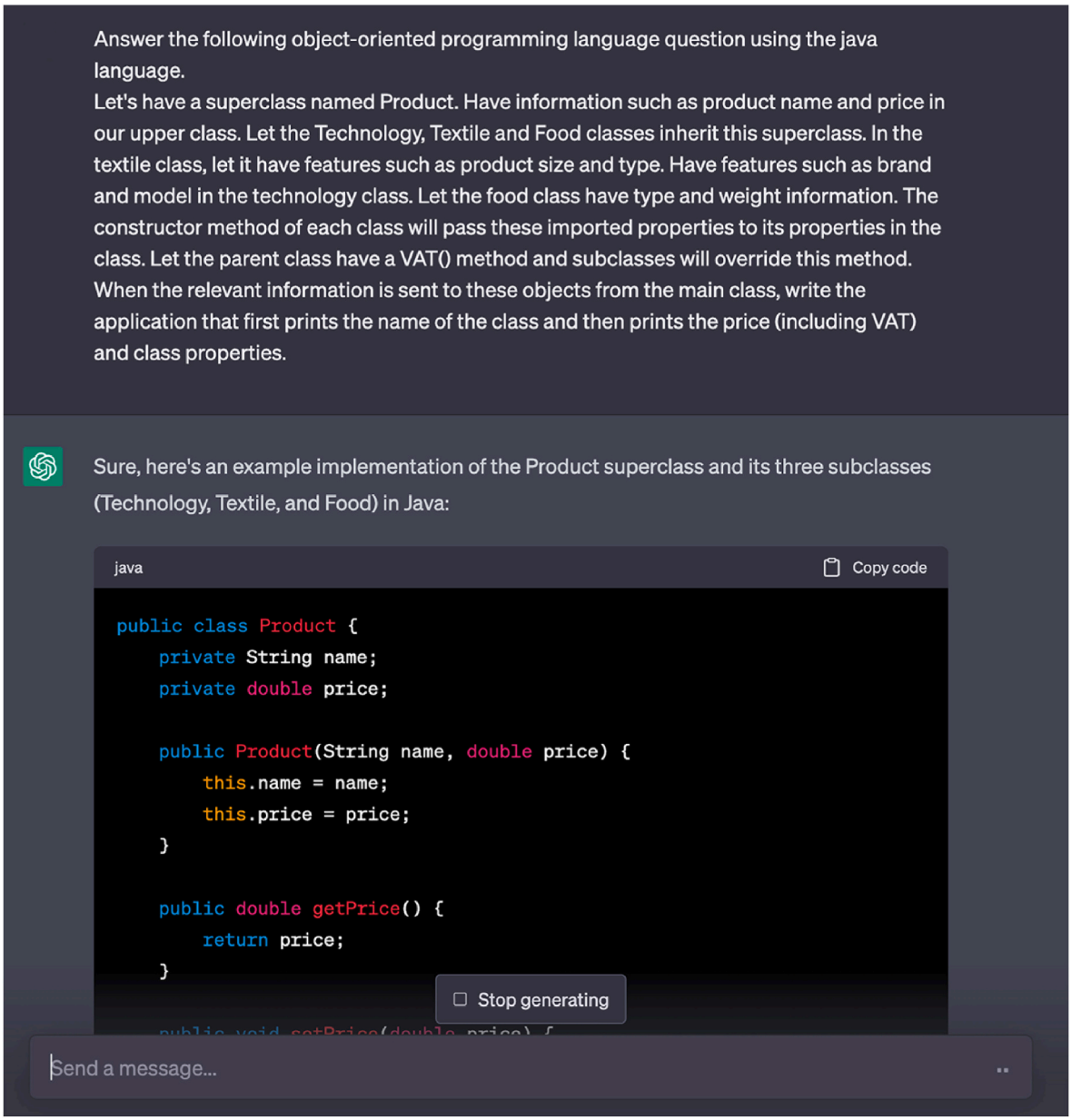

Fig. 3. ChatGPT's response to the desired complex programming application.

SD = 1.96) were significantly higher than those of the control group (M = 32.35, SD = 4.30) with a medium effect size of [$F_{(1,42)}$: 31.359, p = .000, Cohen's f = 0.427] (Cohen, 1992). Similarly, the experimental group's algorithmic thinking posttest scores (M = 24.95, SD = 2.87) were significantly higher than those of the control group (M = 21.83, SD = 4.2) with a small effect size of [$F_{(1,42)}$: 8.187, p = .007, Cohen's f = 0.163] (Cohen, 1992). The experimental group also had significantly higher posttest scores for cooperativity (M = 18.09, SD = 1.82) and critical thinking (M = 21.55, SD = 2.84) compared to the control group, with small effect sizes of [$F_{(1,42)}$: 4.786, p = .034, Cohen's f = 0.102] and [$F_{(1,42)}$: 4.765, p = .035, Cohen's f = 0.102], respectively. Lastly, the experimental group's problem-solving posttest scores (M = 25.00, SD = 2.99) were significantly higher than those of the control group (M = 22.26, SD = 4.09) with a small effect size of [$F_{(1,42)}$: 7.449, p = .009, Cohen's f = 0.151] Cohen, 1992).

3.2. Findings on computer programming self-efficacy

Table 3 provides the descriptive statistics for the computer programming self-efficacy of the study groups.

The equivalency of students' computer programming self-efficacy pretest scores can be observed in Table 3. To address the second research question and its sub-questions, an ANCOVA analysis was conducted after verifying that the assumptions were met for the computer programming self-efficacy scale. The results of this analysis can be found in Table 4.

Based on the results presented in Table 4, the experimental group demonstrated significantly higher computer programming self-efficacy posttest scores (M = 41.32; SD = 5.84) compared to the control group (M = 33.52; SD = 7.64), with a medium effect size [$F_{(1,42)}$: 15.144; p = .000; Cohen's f = 0.265] (Cohen, 1992). Additionally, the experimental group also scored significantly higher on simple programming tasks posttest scores (M = 18.18; SD = 3.58) compared to the control group (M = 14.43; SD = 4.21), with a medium effect size [$F_{(1,42)}$: 12.097; p = .001; Cohen's f = 0.224] (Cohen, 1992). On the other hand, for complex programming tasks scores, the experimental group (M = 23.14; SD = 4.45) scored significantly higher compared to the control group (M = 19.09; SD = 5.04) with a small effect size [$F_{(1,42)}$: 8.210; p = .006;

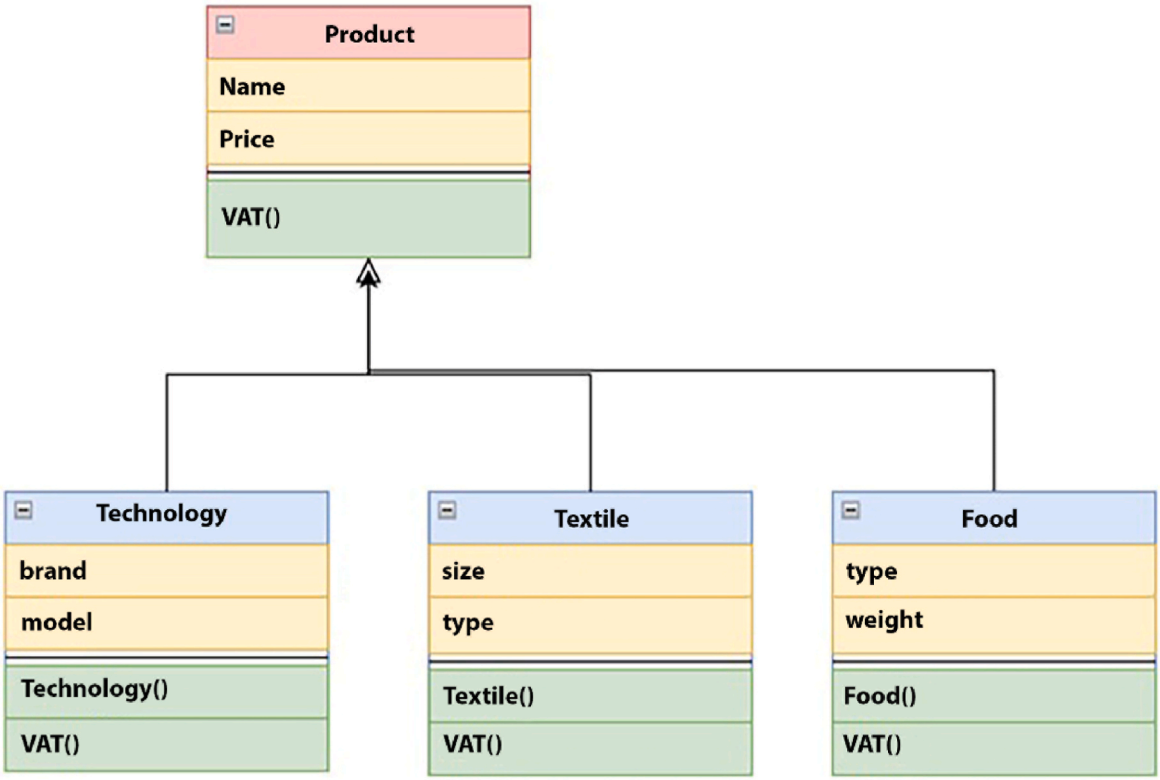

Fig. 4. An example from the weekly lab assignment UML diagram.

Cohen's f = 0.164] (Cohen, 1992).

3.3. Findings on learning motivation

Table 5 provides the descriptive statistics for the learning motivation of the study groups.

Table 5 indicates that the pretest scores for student learning motivation were similar across the groups. In order to investigate the third research question and its corresponding sub-questions, a covariance analysis was performed. Prior to conducting the ANCOVA analysis, the assumptions were tested and found to be met for the learning motivation scale. The findings of the ANCOVA analysis are presented in Table 6.

Table 6 presents the results of the ANCOVA analysis for the posttest scores on various measures of learning motivation. The experimental group demonstrated significantly higher scores on the learning motivation scale (M = 96.50; SD = 7.62) compared to the control group (M = 85.17; SD = 13.84) with a medium effect size, [$F_{(1,42)}$: 11.412; p = .002; Cohen's f = 0.214] (Cohen, 1992).

Regarding the sub-questions, the experimental group had significantly higher scores on individual attitude and expectation (M = 22.09; SD = 1.77) compared to the control group (M = 19.70; SD = 3.07) with a small effect size, [$F_{(1,42)}$: 9.941; p = .003; Cohen's f = 0.191] (Cohen, 1992). The experimental group also had significantly higher scores on clear direction (M = 16.64; SD = 1.65) and reward and recognition (M = 15.86; SD = 2.08) compared to the control group with small effect sizes, [$F_{(1,42)}$: 5.729; p = .021; Cohen's f = 0.120] and [$F_{(1,42)}$: 4.365; p = .043; Cohen's f = 0.094], respectively (Cohen, 1992).

However, the experimental group did not demonstrate a significant difference in challenging goals scores (M = 14.41; SD = 2.75) compared to the control group (M = 14.83; SD = 2.92), [$F_{(1,42)}$: 0.502; p = .482]. The experimental group had significantly higher scores on punishment (M = 8.45; SD = 3.05) and social pressure and competition (M = 19.05; SD = 3.99) compared to the control group with small effect sizes, [$F_{(1,42)}$: 7.241; p = .010; Cohen's f = 0.147] and [$F_{(1,42)}$: 5.610; p = .023; Cohen's f = 0.118], respectively (Cohen, 1992).

4. Discussion

In this research, which was carried out according to the experimental design to determine the effectiveness of AI-supported programming education, AI support was given to the experimental group by using the ChatGPT tool. The control group received traditional programming education. In the study, computational thinking skills, programming self-efficacy, and motivation for the lesson scores of the students in the experimental and control groups were compared. The findings obtained from the research are discussed below.

In the first sub-problem of the study, it was examined whether the use of ChatGPT made a significant difference on the computational thinking skills of the experimental and control group students. The findings of the research showed that the use of ChatGPT significantly increased the computational thinking skills of the students. It was understood from the observations made during the application process that the experimental group students who want to make the most effective use of the ChatGPT tool in the process of making laboratory applications should first follow an algorithm to solve the problem, determine the subprogram particles in line with this algorithm and ask the most appropriate question. Students took the codes of subprogram fragments from the ChatGPT and combined them and tried to reach the desired output. It was seen that this process improves students' computational thinking skills. In other words, instead of spending time writing code, students devoted their time to creative thinking, asking original questions, algorithmic thinking, problem-solving, and critical thinking. As a result, the student can reach the answer about the code snippets they want by asking the most appropriate questions to ChatGPT. The students in the control group, on the other hand, devoted time to processes such as writing code, debugging, and integrating, as well as spending time on thinking processes, which is one of the difficult aspects of programming education. When the literature was examined, it was seen that there was

Table 1
The descriptive statistics computational thinking skills of study groups.

Scales	Pre-post tests	Groups	Mean ($\bar{x}$)	sd
Computational thinking skills	Pretest	Experimental group	110.82	11.21
		Control group	108.48	17.82
	Posttest	Experimental group	126.73	8.34
		Control group	112.61	15.32
Creativity	Pretest	Experimental group	33.23	3.21
		Control group	32.04	6.14
	Posttest	Experimental group	37.14	1.96
		Control group	32.35	4.30
Algorithmic thinking	Pretest	Experimental group	20.73	3.53
		Control group	20.00	3.61
	Posttest	Experimental group	24.95	2.87
		Control group	21.83	4.20
Cooperativity	Pretest	Experimental group	16.05	3.46
		Control group	16.00	2.98
	Posttest	Experimental group	18.09	1.82
		Control group	16.57	2.97
Critical thinking	Pretest	Experimental group	19.14	3.12
		Control group	19.17	2.95
	Posttest	Experimental group	21.55	2.84
		Control group	18.96	5.43
Problem solving	Pretest	Experimental group	21.68	2.42
		Control group	21.26	6.29
	Posttest	Experimental group	25.00	2.99
		Control group	22.26	4.09

no research examining the effect of the use of language models such as ChatGPT on students' computational thinking skills. However, it was seen that there were various studies examining the effect of the use of various AI tools on students' computational thinking skills. Lin and Chen (2020) found that the computational thinking skills of students who used the deep learning recommendation-based system in programming education were significantly higher than those who used the non-deep learning recommendation-based system. Huang and Qiao (2022) found that experimental group students who received AI training with STEAM model had significantly higher computational thinking skills than control group students who did not receive this training. Hsu et al. (2023) found that the use of voice assistant in the course had a significant effect on improving students' computational thinking skills. García et al. (2019) found that providing machine learning and AI training to students was effective in improving students' computational thinking skills. When the results were analyzed in general, it can be said that providing AI training to students and using AI-supported tools were effective in improving students' computational thinking skills. In this direction, it can be concluded that using ChatGPT in programming education was effective in improving students' computational thinking skills.

In the second sub-problem of the study, it was examined whether the use of ChatGPT made a significant difference on the programming self-efficacy of the experimental and control group students. The findings of

Table 3
The descriptive statistics of computer programming self-efficacy of study groups.

Scales	Pre-post tests	Groups	Mean ($\bar{x}$)	sd
Computer programming self-efficacy	Pretest	Experimental group	31.86	13.04
		Control group	32.74	10.75
	Posttest	Experimental group	41.32	5.84
		Control group	33.52	7.64
Simple programming tasks	Pretest	Experimental group	12.41	5.80
		Control group	12.78	4.93
	Posttest	Experimental group	18.18	3.58
		Control group	14.43	4.21
Complex programming tasks	Pretest	Experimental group	19.45	9.72
		Control group	19.96	7.03
	Posttest	Experimental group	23.14	4.45
		Control group	19.09	5.04

Table 2
The results of covariance analysis of computational thinking skills.

	Source	Sum of Squares	df	Mean Square	F	p	Partial Eta Squared
Computational thinking skills	Pretest	2438.558	1	2438.558	24.448	.000	
	Group	1871.185	1	1871.185	18.760	.000	.309
	Error	4189.284	42	99.745			
	Total	8869.244	44				
Creativity	Pretest	220.489	1	220.489	34.642	.000	
	Group	199.589	1	199.589	31.359	.000	.427
	Error	267.319	42	6.365			
	Total	745.644	44				
Algorithmic thinking	Pretest	113.727	1	113.727	10.697	.002	
	Group	87.040	1	87.040	8.187	.007	.163
	Error	446.532	42	10.632			
	Total	670.311	44				
Cooperativity	Pretest	37.759	1	37.759	7.026	.011	
	Group	25.721	1	25.721	4.786	.034	.102
	Error	225.712	42	5.374			
	Total	289.644	44				
Critical thinking	Pretest	142.523	1	142.523	8.856	.005	
	Group	76.681	1	76.681	4.765	.035	.102
	Error	675.888	42	16.093			
	Total	893.778	44				
Problem solving	Pretest	133.721	1	133.721	13.286	.001	
	Group	74.969	1	74.969	7.449	.009	.151
	Error	422.714	42	10.065			
	Total	640.800	44				

Table 4
The results of covariance analysis of computer programming self-efficacy.

	Source	Sum of Squares	df	Mean Square	F	p	Partial Eta Squared
Computer programming self-efficacy	Pretest	65.400	1	65.400	1.418	.240	
	Group	698.478	1	698.478	15.144	.000	.265
	Error	1937.112	42	46.122			
	Total	2686.000	44				
Simple programming tasks	Pretest	82.910	1	82.910	6.045	.018	
	Group	165.911	1	165.911	12.097	.001	.224
	Error	576.015	42	13.715			
	Total	816.800	44				
Complex programming tasks	Pretest	15.429	1	15.429	.676	.416	
	Group	187.464	1	187.464	8.210	.006	.164
	Error	958.988	42	22.833			
	Total	1158.800	44				

Table 5
The descriptive statistics of learning motivation of study groups.

Scales	Pre-post tests	Groups	Mean ($\bar{x}$)	sd
Learning motivation	Pretest	Experimental group	84.32	15.09
		Control group	83.65	10.97
	Posttest	Experimental group	96.50	7.62
		Control group	85.17	13.84
Individual attitude and expectation	Pretest	Experimental group	19.09	4.01
		Control group	19.04	2.70
	Posttest	Experimental group	22.09	1.77
		Control group	19.70	3.07
Challenging goals	Pretest	Experimental group	14.18	2.58
		Control group	14.04	2.44
	Posttest	Experimental group	14.41	2.75
		Control group	14.83	2.92
Clear direction	Pretest	Experimental group	14.73	3.01
		Control group	14.65	1.61
	Posttest	Experimental group	16.64	1.65
		Control group	14.96	2.85
Reward and recognition	Pretest	Experimental group	14.68	2.61
		Control group	14.43	2.73
	Posttest	Experimental group	15.86	2.08
		Control group	14.22	3.09
Punishment	Pretest	Experimental group	5.77	2.78
		Control group	5.91	3.10
	Posttest	Experimental group	8.45	3.05
		Control group	5.87	3.29
Social pressure and competition	Pretest	Experimental group	15.86	6.04
		Control group	15.57	4.26
	Posttest	Experimental group	19.05	3.99
		Control group	15.61	5.46

the research showed that the use of ChatGPT significantly increased students' programming self-efficacy. Thanks to the advantages such as coding and debugging provided by AI support, it was determined that the student's self-efficacy in the experimental group regarding programming and their motivation towards the lesson improved significantly compared to the students in the control group. In other words, the advantages that ChatGPT provided in the coding process enabled the students to develop their coding-related self-efficacy. When the literature was examined, no research examining the effect of using ChatGPT on students' programming self-efficacy was found. However, it has been observed that there are various research results examining the effectiveness of using AI tools on students' self-efficacy. Huang and Qiao (2022) found that experimental group students who received AI training with STEAM model had significantly higher self-efficacy than control group students who did not receive this training. According to Li and Wang (2021) research, artificial intelligence capability in higher education institutions has been proven to positively affect students' creativity and self-efficacy in learning performance. Wang, Sun, and Chen (2022) revealed that higher education institutes' artificial intelligence capability directly affects self-efficacy. It can be said that these results support the results of our research. Therefore, it can be stated that the use of AI tools such as ChatGPT in programming education is effective in increasing students' programming self-efficacy.

In the third sub-problem of the study, it was examined whether the use of ChatGPT made a significant difference on the motivation of the experimental and control group students. The research findings showed that the use of ChatGPT significantly increased the motivation of the students. When the literature was examined, no research examining the effect of ChatGPT use on students' motivation was found. However, it was seen that there are various research results examining the effectiveness of using AI tools on students' motivation. Huang and Qiao (2022) found that the experimental group students who received AI training with STEAM model had significantly higher motivation than the control group students who did not receive this training. Sharma et al. (2020) found that the use of eye-tracking and AI tools in the lesson increased students' motivation to learn. Huang et al. (2023) found that AI-enabled personalized video recommendations can significantly improve the earning performance and engagement of students with a moderate level of motivation. Based on these results, it can be stated that using AI tools such as ChatGPT was effective in increasing students' self-efficacy towards programming course.

The results related to the sub-problems of our research (computational thinking skills, self-efficacy, motivation) were presented above. In addition to this, the results of our research and studies that generally address the learning outcomes of students with and without the use of AI tools (e.g. Chatbot etc.) are discussed below. When the literature was examined, it was seen that no research results examined the effect of the ChatGPT use on students' learning outcomes. However, it is seen that there is various research other than programming education for the use of educational intelligent teaching systems, learning analytics, adaptive and recommender systems, and chatbots. Figueiredo and García-Peñalvo (2020b) investigated the effectiveness of using intelligent tutoring systems in programming education. As a result of the research, it was concluded that the use of intelligent tutoring systems had a beneficial effect on students' success scores, pass/fail rates, students' interest and participation in the course, problem-solving skills, motivation, and passion. According to Yilmaz et al. (2022), it was revealed that the Smart MOOC system is beneficial in terms of providing personalized feedback to the student, predicting the learning performance of the student and making recommendations, and improving students' motivation, self-assessment skills, and self-efficacy. Yin et al. (2021), in their

Table 6
The results of covariance analysis of metacognitive awareness.

	Source	Sum of Squares	df	Mean Square	F	p	Partial Eta Squared
Learning motivation	Pretest	236.671	1	236.671	1.914	.174	
	Group	1411.371	1	1411.371	11.412	.002	.214
	Error	5194.134	42	123.670			
	Total	6873.244	44				
Individual attitude and expectation	Pretest	.459	1	.459	.071	.792	
	Group	64.432	1	64.432	9.941	.003	.191
	Error	272.229	42	6.482			
	Total	337.200	44				
Challenging goals	Pretest	107.624	1	107.624	18.913	.000	
	Group	2.857	1	2.857	.502	.482	
	Error	238.999	42	5.690			
	Total	348.578	44				
Clear direction	Pretest	6.946	1	6.946	1.273	.266	
	Group	31.249	1	31.249	5.729	.021	.120
	Error	229.101	42	5.455			
	Total	267.778	44				
Reward and recognition	Pretest	1.598	1	1.598	.225	.638	
	Group	31.068	1	31.068	4.365	.043	.094
	Error	298.906	42	7.117			
	Total	330.978	44				
Punishment	Pretest	1.807	1	1.807	.176	.677	
	Group	74.526	1	74.526	7.241	.010	.147
	Error	432.256	42	10.292			
	Total	509.200	44				
Social pressure and competition	Pretest	18.585	1	18.585	.803	.375	
	Group	129.805	1	129.805	5.610	.023	.118
	Error	971.848	42	23.139			
	Total	1123.244	44				

research on university students within the scope of basic computer science courses, concluded that chatbot-based learning effectively increases students' motivation. Chang et al. (2022), in their research on nursing students, revealed that mobile chatbot applications could increase the learning success and self-efficacy of nursing students. Lee et al. (2022) showed that the application of artificial intelligence-based chatbots in the review process of public health courses could increase students' academic performance, self-efficacy, learning attitude, and motivation. According to Katchapakirin et al. (2022), they developed ScratchThAI, a chatbot developed for Scratch, a block-based programming language for young learners. Various elements, such as gamification, have been added to the developed tool to improve the computational thinking skills of the students and enhance their motivation. It is stated that the developed system enhances teacher satisfaction, better learning performance, and higher student participation. Fryer et al. (2019) showed that using chatbots in language learning increases the interest and participation of learners. Huang et al. (2022) revealed that using chatbots in language learning can help provide feedback and increase student interest, participation, and satisfaction. It is seen that the findings in the literature on the effects of chatbot use on students' learning process and results are generally like the results of our study. However, considering the advanced language model features of the chatbot (ChatGPT) used in our research, it can be stated that it can produce more meaningful student results.

Based on the research process and findings, we can put forward some inferences and suggestions for researchers and educators. First of all, it should be taken into consideration that the ChatGPT is a text-based chatbot and can give instant answers, the language library is quite advanced for different languages, such as Turkish, and it can develop much better over time. Therefore, this tool can give mostly correct and logical answers to text-based questions. Therefore, we faced the reality that students can do their homework, especially in fields such as social sciences, with this tool. In this case, what teachers should do is not give assignments/questions so that they can get answers by asking the questions given to ChatGPT. For example, this tool can even respond to complex programming assignments given within the scope of this research. In order to prevent this, the assignments are expressed visually, as shown in Fig. 3, and it is seen that homework is not a single solution. It is aimed at enabling students to think differently. Therefore, the student needs to advance their thinking skills on the subject first to know what to ask in ChatGPT.

For courses such as programming education, there is a process where it is more important to be able to apply than to know. Because the student can get the desired answer from this tool. The important thing for the student who gets the code snippet she/he wants from this tool is how she/he will integrate it into the whole program, whether she/he can run the program or not. For this reason, it is very important to provide students with practical skills as well as theoretical knowledge. In order to do this, it is important to use methods such as project-based learning and collaborative learning in the teaching process. At the end of the semester, the student will be able to learn how to use AI tools in the process of presenting the project given to them as a whole, and how to integrate the sub-work packages of the project to reveal the final product. In other words, they will be able to put what they know into practice. In order for students to benefit from this tool effectively, it is important that their thinking skills and imagination are developed. Considering that these skills are developmental skills, it would be beneficial to make revisions/improvements in the curricula aimed at gaining these skills from an early age.

This study has some limitations. First, the implementation process of the study lasted five weeks. In future studies, the effectiveness of ChatGPT can be examined by conducting longer studies such as longitudinal studies. Another limitation of the study is that the experimental process was conducted with 45 students. In further research, the number of participants can be increased and the results can be compared. Within the scope of the research, programming tasks were given as UML diagrams. In forthcoming investigations, the diversity of programming tasks can be increased by presenting different case scenarios. In this study, students performed individual programming tasks. Further research can examine the effectiveness of using ChatGPT in collaborative or group-based learning activities that simulate real IT workplace environments on individual learning outcomes and group collaboration processes. Again, in other future studies, the effectiveness of using ChatGPT individually and in collaborative groups such as pair-

programming groups can be compared.

5. Conclusion

In this study, the effect of ChatGPT-supported programming education on computational thinking, programming self-efficacy, and motivation for the programming course of university students was examined. The research was carried out according to the experimental design with pretest-posttest control group, while the experimental group benefited from ChatGPT in the programming learning process, the control group students did not use this tool. As a result of the research, it was determined that the use of ChatGPT in programming education statistically significantly increased students' computational thinking skills, programming self-efficacy and motivation for the lesson. In terms of the sub-dimensions of the scales, it was concluded that the use of ChatGPT created significant differences in the dimensions of creativity, algorithmic thinking, cooperativity, critical thinking, problem solving, simple programming tasks, complex programming tasks, individual attitude and expectation, clear direction, reward and recognition, punishment, social pressure and competition. However, in the sub-dimension of challenging goals, it was concluded that there was no significant difference between the experimental and control groups. The challenging goals sub-dimension is related to students' motivation when they encounter challenging problems in programming assignments. Therefore, when students are given challenging problems, the use of AI tools such as ChatGPT does not have a significant effect on increasing student motivation. For this reason, it is important for teachers to seek various motivational strategies to ensure student motivation in the face of challenging tasks. When the results obtained from the scales and its sub-dimensions are evaluated in general, it can be stated that using AI tools and environments such as ChatGPT in programming education was beneficial for students' learning process and outcomes.

In order for students to benefit most effectively from AI tools and environments such as ChatGPT, it is important to provide students with prompt writing skills. Being able to write effective prompts will enable students to use tools such as ChatGPT effectively and efficiently. When integrating tools such as ChatGPT into their lessons, it is recommended that teachers gain AI literacy skills, especially for students to gain prompt writing skills. Teachers can apply metacognitive strategies when it is considered important that students' thinking skills are developed in order to benefit effectively from tools such as AI. At this point, metacognitive prompts can be used. The metacognitive prompt aims to enable students to think and evaluate their own learning processes. Such teaching strategies help students understand, control and regulate their own learning processes. For example, the teacher asked, "What questions do you need to ask to solve this problem?", "What kind of question can you ask in order to produce a more original solution to the problem?" By asking such questions, students can question their own thoughts and direct their own learning processes.

Ethical declarations

Prior to data collection, each participant was asked to read an information sheet outlining the study. They were asked to agree with a written declaration of informed consent. Bartın University Scientific Research and Publication Ethics Guidelines were complied with for this study.

Ethical approval

All procedures performed in these studies were in accordance with the APA ethical guidelines, the ethical standards of the institutional research committee, and the 1964 Helsinki declaration and its later amendments.

Data availability

The authors are willing to share their data, analytics methods, and study materials with other researchers upon request.

Code availability

The authors used AMOS functions for their statistical analyses.

Informed consent to participate

All participants gave full informed consent to participate.

Consent for publication

All participants gave consent for their data to be used in publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

List of acronyms

AI	Artificial Intelligence
GPT	Generative Pre-trained Transformer
UML:	Unified Modeling Language

Appendix 1. Computational thinking scale items

Creativity

I like the people who are sure of most of their decisions.
I like the people who are realistic and neutral.
I believe that I can solve most of the problems I face if I have sufficient amount of time and if I show effort.
I have a belief that I can solve the problems possible to occur when I encounter with a new situation.
I trust that I can apply the plan while making it to solve a problem of mine.
Dreaming causes my most important projects to come to light.
I trust my intuitions and feelings of "trueness" and "wrongness" when I approach the solution of a problem.
When I encounter with a problem, I stop before proceeding to another subject and think over that problem.

Algoritmic thinking

I can immediately establish the equity that will give the solution of a problem.
I think that I have a special interest in the mathematical processes.
I think that I learn better the instructions made with the help of mathematical symbols and concepts.
I believe that I can easily catch the relation between the figures.
I can mathematically express the solution ways of the problems I face in the daily life.
I can digitize a mathematical problem expressed verbally.

Cooperativity

I like experiencing cooperative learning together with my group friends.
In the cooperative learning, I think that I attain/will attain more successful results because I am working in a group.
I like solving problems related to group project together with my friends in cooperative learning.
More ideas occur in cooperative learning.

Critical thinking

I am good at preparing regular plans regarding the solution of the complex problems.
It is fun to try to solve the complex problems.
I am willing to learn challenging things.
I am proud of being able to think with a great precision.
I make use of a systematic method while comparing the options at my hand and while reaching a decision.

Problem solving

I have problems in the demonstration of the solution of a problem in my mind.
I have problems in the issue of where and how I should use the variables such as X and Y in the solution of a problem.
I cannot apply the solution ways I plan respectively and gradually.
I cannot produce so many options while thinking of the possible solution ways regarding a problem.
I cannot develop my own ideas in the environment of cooperative learning.
It tires me to try to learn something together with my group friends in cooperative learning.

Appendix 2. ChatGPT's response to the desired simple programming application

```
import java.util.Scanner;

public class AverageCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the first number:");
        double num1 = scanner.nextDouble();

        System.out.println("Enter the second number:");
        double num2 = scanner.nextDouble();

        double average = (num1 + num2) / 2.0;

        System.out.println("The average of " + num1 + " and " + num2 + " is " + average);
    }
}
```

Appendix 3. ChatGPT's response to the desired complex programming application

```
public class Product {
    private String name;
    private double price;

    public Product(String name, double price) {
        this.name = name;
        this.price = price;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public double VAT() {
        return price * 0.18;
    }
}

public class Technology extends Product {
    private String brand;
    private String model;

    public Technology(String name, double price, String brand, String model) {
        super(name, price);
        this.brand = brand;
        this.model = model;
    }

    @Override
    public double VAT() {
        return super.VAT() * 1.1;
    }
}

public class Textile extends Product {
    private String size;
    private String type;

    public Textile(String name, double price, String size, String type) {
        super(name, price);
        this.size = size;
        this.type = type;
    }

    @Override
    public double VAT() {
        return super.VAT() * 1.08;
    }
}

public class Food extends Product {
    private String type;
    private double weight;

    public Food(String name, double price, String type, double weight) {
        super(name, price);
        this.type = type;
        this.weight = weight;
    }

    @Override
    public double VAT() {
        return super.VAT() * 1.05;
    }
}
```

References

Alam, A. (2022). Educational robotics and computer programming in early childhood education: A conceptual framework for assessing elementary school students' computational thinking for designing powerful educational scenarios. In *2022 international conference on Smart Technologies and Systems for next generation computing (ICSTSN)* (pp. 1–7). IEEE.

Altun, A., & Mazman, S. G. (2012). The validity and reliability study of the Turkish version of the self-efficacy perception regarding programming scale. *Journal of Measurement and Evaluation in Education and Psychology, 3*(2), 297–308.

Avci, U., & Ersoy, H. (2018). The adaptation of learning motivation in computer programming courses scale into Turkish: The study of validity and reliability. *Journal of Higher Education and Science, 8*(1), 73–81.

Chang, C. Y., Hwang, G. J., & Gau, M. L. (2022). Promoting students' learning achievement and self-efficacy: A mobile chatbot approach for nursing training. *British Journal of Educational Technology, 53*(1), 171–188.

Cohen, J. (1992). Statistical power analysis. *Current Directions in Psychological Science, 1*(3), 98–101.

Eteng, I., Akpotuzor, S., Akinola, S. O., & Agbonlahor, I. (2022). A review on effective approach to teaching computer programming to undergraduates in developing countries. *Scientific African*, Article e01240.

Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2021). Computational thinking in programming with scratch in primary schools: A systematic review. *Computer Applications in Engineering Education, 29*(1), 12–28.

Figueiredo, J., & García-Peñalvo, F. J. (2020). Increasing student motivation in computer programming with gamification. In *In2020 IEEE global engineering education conference (EDUCON)* (pp. 997–1000). IEEE.

Figueiredo, J., & García-Peñalvo, F. J. (2020). Intelligent tutoring systems approach to introductory programming courses. *InEighth International Conference on Technological Ecosystems for Enhancing Multiculturality*, 34–39.

Fryer, L. K., Nakao, K., & Thompson, A. (2019). Chatbot learning partners: Connecting learning experiences, interest and competence. *Computers in Human Behavior, 93*, 279–289.

García, J. D. R., León, J. M., González, M. R., & Robles, G. (2019). Developing computational thinking at school with machine learning: An exploration. In *In2019 international symposium on computers in education (SIIE)* (pp. 1–6). IEEE.

González-Pérez, L. I., & Ramírez-Montoya, M. S. (2022). Components of education 4.0 in 21st century skills frameworks: Systematic review. *Sustainability, 14*(3), 1493.

Handur, V., Kalwad, P. D., Patil, M. S., Garagad, V. G., Yeligar, N., Pattar, P., … Joshi, G. H. (2016). Integrating class and laboratory with hands-on programming: Its benefits and challenges. In *In2016 IEEE 4th international conference on MOOCs, innovation and technology in education (MITE)* (pp. 163–168). IEEE.

Hsu, T. C., Chang, C., & Lin, Y. W. (2023). Effects of voice assistant creation using different learning approaches on performance of computational thinking. *Computers & Education, 192*, Article 104657.

Huang, A. Y., Lu, O. H., & Yang, S. J. (2023). Effects of artificial Intelligence–Enabled personalized recommendations on learners' learning engagement, motivation, and outcomes in a flipped classroom. *Computers & Education, 194*, Article 104684.

Huang, X., & Qiao, C. (2022). *Enhancing computational thinking skills through artificial intelligence education at a STEAM high school* (pp. 1–21). Science & Education. https://doi.org/10.1007/s11191-022-00392-6

James, J. (2021). Confronting the scarcity of digital skills among the poor in developing countries. *Development Policy Review, 39*(2), 324–339.

Karaoglan Yilmaz, F. G., & Yilmaz, R. (2022a). Examining student satisfaction with the use of smart mooc. In *International i?stanbul scientific research congress*. Turkey: Istanbul. July 23-25, 2022.

Karaoglan Yilmaz, F. G., & Yilmaz, R. (2022b). Examining student views on the use of the learning analytics dashboard of a smart mooc. In *International Azerbaijan congress on life, social, health, and art sciences*. August 13-14, 2022, Azerbaijan.

Kasneci, E., Seßler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., … Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences, 103*, Article 102274.

Katchapakirin, K., Anutariya, C., & Supnithi, T. (2022). ScratchThAI: A conversation-based learning support framework for computational thinking development. *Education and Information Technologies, 27*(6), 8533–8560.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior, 72*, 558–569.

Law, K. M. Y., Lee, V. C. S., & Yu, Y. T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education, 55*, 218–228.

Lee, Y. F., Hwang, G. J., & Chen, P. Y. (2022). Impacts of an AI-based chat bot on college students' after-class review, academic performance, self-efficacy, learning attitude, and motivation. *Educational Technology Research & Development, 70*(5), 1843–1865.

Lin, P. H., & Chen, S. Y. (2020). Design and evaluation of a deep learning recommendation based augmented reality system for teaching programming and computational thinking. *IEEE Access, 8*, 45689–45699.

Li, Z., & Wang, H. (2021). The effectiveness of physical education teaching in college based on Artificial intelligence methods. *Journal of Intelligent and Fuzzy Systems, 40*(2), 3301–3311.

Liu, J., Sun, M., Dong, Y., Xu, F., Sun, X., & Zhou, Y. (2022a). The mediating effect of creativity on the relationship between mathematic achievement and programming self-efficacy. *Frontiers in Psychology, 12*, 6243.

Liu, H., Wu, Z., Lu, Y., & Zhu, L. (2022b). Exploring the balance between computational thinking and learning motivation in elementary programming education: An empirical study with game-based learning. *IEEE Transactions on Games, 15*(1), 95–107.

Lo, C. K. (2023). What is the impact of ChatGPT on education? A rapid review of the literature. *Education Sciences, 13*(4), 410.

López-Pimentel, J. C., Medina-Santiago, A., Alcaraz-Rivera, M., & Del-Valle-Soto, C. (2021). Sustainable project-based learning methodology adaptable to technological advances for web programming. *Sustainability, 13*(15), 8482.

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

Malik, S., Al-Emran, M., Mathew, R., Tawafak, R., & AlFarsi, G. (2020). Comparison of E-learning, M-learning and game-based learning in programming education–a gendered analysis. *International Journal of Emerging Technologies in Learning (iJET), 15*(15), 133–146.

Mathew, R., Malik, S. I., & Tawafak, R. M. (2019). Teaching problem solving skills using an educational game in a computer programming course. *Informatics in Education, 18*(2), 359–373.

OpenAI. (2023). *ChatGPT*. Retrieved from https://chat.openai.com/chat 05.01.2023.

Ramalingam, V., & Wiedenbeck, S. (1998). Development and validation of scores on a computer programming self efficacy scale and group analyses of novice programmer self-efficacy. *Journal of Educational Computing Research, 19*(4), 365–379.

Sharma, K., Giannakos, M., & Dillenbourg, P. (2020). Eye-tracking and artificial intelligence to enhance motivation and learning. *Smart Learning Environments, 7*(1), 1–19.

Strawhacker, A., & Bers, M. U. (2019). What they learn when they learn coding: Investigating cognitive domains and computer programming knowledge in young children. *Educational Technology Research & Development, 67*, 541–575.

Sullivan, A., & Strawhacker, A. (2021). Screen-free STEAM: Low-cost and hands-on approaches to teaching coding and engineering to young children. In *Embedding STEAM in early childhood education and care* (pp. 87–113). Cham: Springer International Publishing.

Su, Y. S., Shao, M., & Zhao, L. (2022). Effect of mind mapping on creative thinking of children in scratch visual programming education. *Journal of Educational Computing Research, 60*(4), 906–929.

Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature review. *Computers & Education, 162*, Article 104083.

Tlili, A., Shehata, B., Adarkwah, M. A., Bozkurt, A., Hickey, D. T., Huang, R., & Agyemang, B. (2023). What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart Learning Environments, 10*(1), 15.

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.

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.

Wang, S., Sun, Z., & Chen, Y. (2022). Effects of higher education institutes' artificial intelligence capability on students' self-efficacy, creativity and learning performance. *Education and Information Technologies, 28*, 4919–4939.

Wei, X., Lin, L., Meng, N., Tan, W., & Kong, S. C. (2021). The effectiveness of partial pair programming on elementary school students' computational thinking skills and self-efficacy. *Computers & Education, 160*, Article 104023.

Yilmaz, R., & Karaoglan Yilmaz, F. G. (2022a). Investigation of student views on data privacy and ethical use of data in smart learning environments. In *International i? stanbul scientific research congress*. Turkey: Istanbul. July 23-25, 2022.

Yilmaz, R., & Karaoglan Yilmaz, F. G. (2022b). Investigation of students' self-regulation skills, motivation and disorientation in smart mooc. In *International Azerbaijan congress on life, social, health, and art sciences*. August 13-14, 2022, Azerbaijan.

Yilmaz, R., Karaoglan Yilmaz, F. G., & Keser, H. (2020). Vertical versus shared e-leadership approach in online project-based learning: A comparison of self-regulated learning skills, motivation and group collaboration processes. *Journal of Computing in Higher Education, 32*, 628–654.

Yilmaz, R., Yurdugül, H., Yilmaz, F. G. K., Şahin, M., Sulak, S., Aydin, F., … Ömer, O. R. A. L. (2022). Smart MOOC integrated with intelligent tutoring: A system architecture and framework model proposal. *Computers in Education: Artificial Intelligence, 3*, Article 100092.

Yin, J., Goh, T. T., Yang, B., & Xiaobin, Y. (2021). Conversation technology with micro-learning: The impact of chatbot-based learning on students' learning motivation and performance. *Journal of Educational Computing Research, 59*(1), 154–177.