Education and Information Technologies (2021) 26:6069–6095
https://doi.org/10.1007/s10639-021-10596-y

Studying the effects of teaching programming to lower secondary school students with a serious game: a case study with Python and CodeCombat

Chrysoula Kroustalli[1] · **Stelios Xinogalos**[2]

Received: 16 March 2021 / Accepted: 19 May 2021 / Published online: 31 May 2021

Abstract

Serious games, or else educational games,for programming are considered to have a positive impact on learning programming.Specifically, serious games are considered to motivate students and engage themin playing and learning programming. However, more research is required inorder to study their effects in learning programming, as well as their addedvalue in comparison with typical teaching approaches. In this study the effectsof teaching programming to lowersecondary school students with the serious gameCodeCombat and the typical teaching approach are compared. Specifically,fifty-nine students formed an experimental group that was taught programmingwith CodeCombat and a control group that was taught programming throughlecturing and problem solving in Python with a typical programming environment.The study aimed to investigate whether a game like CodeCombat that is based ona text-based real programming language: improvesstudents' performance in basic programming concepts; brings better learning outcomes in comparison with typical teaching methods; engages students'interest. Data was collected through a pre and post test, as well as a surveyprior the intervention and another one based on the Technology Acceptance Modelat the end. It was concluded that the experimental group performed better thanthe control group, but this difference was not found to be statisticallysignificant. CodeCombat was evaluated positively in terms of its perceived easeof use and usefulness, as well as the attitude towards its use. The resultswere neutral in terms of students' behavioral intention to use CodeCombat, butwere positive in using serious games for programming in general.

Keywords Serious games for programming · CodeCombat · Python · Learning effects · Technology Acceptance Model (TAM)

✉ Stelios Xinogalos
stelios@uom.edu.gr

Extended author information available on the last page of the article

1 Introduction

In recent years, new technologies hold an important place in the lives of children. Children from a small age start using computers for playing games and are accustomed to interacting with environments that have appealing graphical user interfaces (Malliarakis et al., 2014a). These facts require changing our teaching methods. Game-based approaches, such as serious games seem to be an appropriate way to improve students' learning achievements.

A survey in the field of serious games has revealed several definitions. All these definitions argue that serious games are games that have a "serious" purpose. But what are games in the first place? Kinzie and Joseph (2008, p.644) based on a review of several definitions, define a game as *"an immersive, voluntary and enjoyable activity in which a challenging goal is pursued according to agreed-upon rules"*. According to Michael and Chen (2005, p.21), serious games are *"games that do not have entertainment, enjoyment, or fun as their primary purpose"*. This primary purpose in several cases has an educational flavor. Nazry et al. (2017) state that serious games create a sense of happiness and this has a positive effect on learning. Moreover, when digital games are used for learning they do not just stimulate students' interest, but also result in their increased cognitive development (Mitchell & Savill-Smith, 2004). Serious games stimulate active learning and provide support for learning cognitive demanding subjects, such as programming.

Teaching programming enhances students' problem solving, as well as planning and organization skills. It encourages experimentation, develops independence and promotes precision and self-discipline (Siegle, 2017). However, learning computer programming is a difficult process and seems to be accompanied with several problems. Research has shown that novice programmers face difficulties when they have to comprehend and even more apply programming concepts that are complex (Xinogalos, 2016). Furthermore, the teaching methods used for introducing novices to programming have an important effect in the educational process (Brusilovsky et al., 1997). Conventional teaching methods (Brusilovsky et al., 1997) and studying approaches are no longer the most suitable for many students (Gomes & Mendes, 2007; Maragkos & Grigoriadou, 2005; Papadakis & Kalogiannakis, 2018; Virvou et al., 2005). Studies have shown that such approaches reduce both the motivation and interest of students in learning programming (Papadakis & Kalogiannakis, 2018). In an attempt to remedy these problems and support the learning of computer programming several serious games, or else educational games, have been designed (Combéfis et al., 2016; Eguíluz et al., 2018; Laporte & Zaman, 2016; Malliarakis et al., 2014b; Miljanovic & Bradbury, 2018; Vahldick et al., 2014).

The majority of the serious games for programming are block-based environments (Eguíluz et al. 2018; Giannakoulas & Xinogalos, 2020; Vahldick et al., 2014) in order to avoid syntax errors and help novices concentrate on comprehending the programming concepts. Text-based serious games for programming also exist, but their impact on learning programming has not been adequately

investigated. Actually, the studies that investigate the impact of serious games for programming in the classroom are few (Giannakoulas & Xinogalos, 2018), no matter if they are block-based or text-based. Since the ultimate goal for secondary education students is to start using a real text-based programming language at some point during their studies (Esper et al., 2014), we consider it important to investigate whether this could be accomplished through a game. Moreover, we consider it important to investigate whether learning a real programming language, such as Python, through a game is more effective than the typical teaching approach based on an integrated programming environment for a conventional language and typical number and symbol processing problems (Brusilovsky et al., 1997).

In order to investigate the aforementioned issues, a didactical intervention based on the serious game CodeCombat was designed for teaching fundamental programming concepts with Python to lower secondary school students. Specifically, an experimental group that used the serious game CodeCombat and a control group that used a conventional Python programming environment were formed. Data collection was accomplished through specially designed pre/post tests, a questionnaire prior the intervention for recording students' perceptions on programming and games, and a questionnaire after the intervention based on the Technology Acceptance Model (TAM) (Davis, 1989; Park, 2009). The research questions (RQs) of the study are:

RQ1: Does the text-based serious game CodeCombat improve lower secondary school students' performance in basic programming concepts?
RQ2: Does the text-based serious game CodeCombat bring better learning outcomes in programming than conventional teaching methods?
RQ3: Does the text-based serious game CodeCombat engage students' interest?

The rest of this article is organized as follows. In Sect. 2, a literature review on serious games for programming and relevant studies is presented. In Sect. 3 the methodology of the study is presented, including information for the teaching intervention, the participants and the data collection instruments and analysis. In Sect. 4 the results are presented and briefly analyzed. In Sect. 5 the main findings are discussed in the context of relevant literature, while in Sect. 6 the limitations, the contribution and the main conclusions of the study are summarized.

2 Literature review

In this section a literature review on serious games for programming is presented focusing on: the distinction of block-based and text-based games; providing the rationale for selecting the text-based game CodeCombat, along with a brief description of the game; the gap in the research on text-based serious games for programming that formulated the research questions of our study.

2.1 Serious games for programming

Serious games for programming have attracted the interest of teachers and researchers for several years. A search in Google Scholar for a (systematic) literature review, overview and/or comparative analysis for serious or educational games for programming revealed several contemporary studies (Combéfis et al., 2016; Eguíluz et al., 2018; Giannakoulas & Xinogalos, 2020; Laporte & Zaman, 2016; Malliarakis et al., 2014b; Miljanovic & Bradbury, 2018; Vahldick et al., 2014). These studies review educational games for programming targeted to students of all ages, attending primary education, secondary education, or even higher education.

A large number of educational games were analyzed in the aforementioned studies and as highlighted most of the games are block-based (Eguíluz et al., 2018; Giannakoulas & Xinogalos, 2020; Vahldick et al., 2014). Block-based games are ideal for introducing young students to programming, since the visual commands are more appealing to students and moreover students avoid syntax errors and can simply concentrate on understanding the semantics of the programming constructs. Games that use a conventional programming language are fewer and their majority utilizes a single programming language (Vahldick et al., 2014). Representative examples of this kind of games are the real time strategy game Colobot (https://colobot.info/) that uses a programming language similar to C++and Java (called CBOT), and the action adventure game CodeSpells (https://codespells.org/) that is based on Java. A widely known educational game that gives students the chance to write code either in Python or JavaScript is CodeCombat (https://codecombat.com/).

Our study aimed to utilize a serious game for supporting the learning of programming concepts by lower secondary education students. Students in Greece are usually introduced to programming in primary school utilizing block-based games, such as Run Marco (Giannakoulas & Xinogalos, 2018), or even more frequently the block-based educational programming environment of Scratch (Sáez-López et al., 2016). Block-based environments are also used for introducing secondary education students to programming concepts. A study applying a game-development approach with the use of MIT App Inventor for teaching programming concepts to secondary education students had a positive effect both on students' achievements and motivation (Papadakis, 2020). Weintrop and Wilensky (2017) in a study comparing the outcomes of block-based and text-based programming environments utilized in a high school introductory programming course concluded that in both environments students were confident, enjoyed the experience and improved their knowledge on programming. However, students using block-based environments exhibited a better performance and increased interest in attending computer courses, while students using text-based environments viewed their experience as being closer to that of professional programmers. Mladenović et al. (2018) concluded that students deal with misconceptions on loops more successfully in block-based environments like Scratch in comparison with text-based environments like Logo and Python. However, students have, at some point in their secondary education studies, to start using a real text-based programming language (Esper et al., 2014). The serious game CodeCombat was considered ideal for this purpose and was selected for our study, since it supports program development with Python that

is proposed for secondary education in Greece and it does so in a gaming environment that visualizes concepts and has the potential to motivate students.

2.2 The serious game CodeCombat

CodeCombat (Fig. 1) is targeted to students aged 9 and up and it has been translated into over 50 languages. As already mentioned, it supports Python and JavaScript and it can be used online, as long as a browser and Internet access are available (https://www.codecombat.com). CodeCombat is a complete programming learning management environment suitable both for beginners and students with prior programming experience. It is also considered suitable for teachers, as it allows the creation of a classroom with the ability to control and monitor student progress.

The first course in CodeCombat, entitled *Introduction to Computer Science*, focuses on teaching fundamental programming concepts. The player can choose his/her avatar, called hero, as well as the programming language that will be used. The available programming languages are Python, JavaScript, CoffeeScript and C++. In each level, the player writes code for guiding the hero in accomplishing various tasks, such as collecting gems, defeating enemies and escaping from a level without touching the spikes or being spotted by ogres. Diamonds collected in one level can be used between levels for acquiring better armor or magic devices that allow for controlling increasingly difficult tasks. If the code is wrong the hero losses points, while plenty of hints are available for guiding the player through the game.

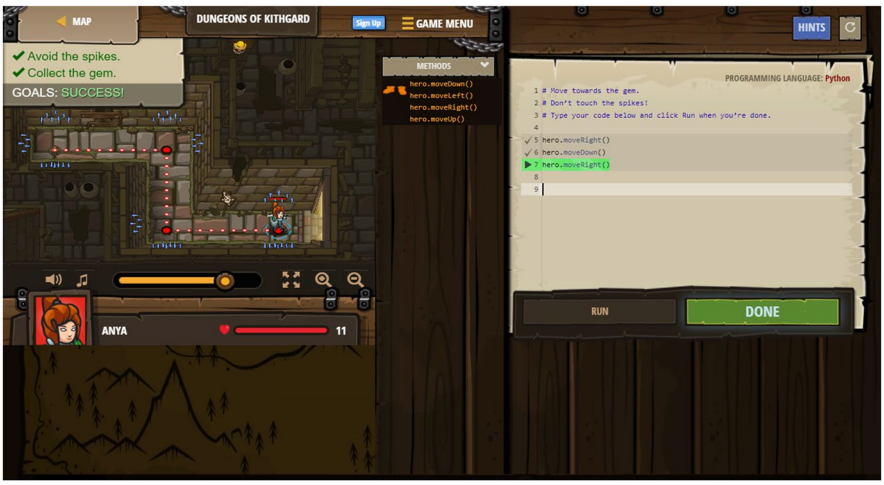

Fig. 1 Screenshot from CodeCombat

2.3 Empirical studies on text-based serious games for programming

Several serious games for programming are available nowadays, but research on their actual effectiveness when used in the classroom is still limited. Based on Miljanovic and Bradbury (2018), evaluation results were recorded for thirty-six out of the forty-nine games reviewed. In most cases the evaluation was based on a survey, while fewer studies utilized game play statistics and skill tests. The most common research questions focused on: users' feelings about the game; accessibility; engagement during game play; and learning effect. The results in most cases were positive.

CodeCombat that was selected for our study was included in the review by Miljanovic and Bradbury (2018), but no evaluation results were recorded. In a Google Scholar search for studies on using CodeCombat for an introduction to programming we were able to find just two studies (Danks et al., 2019; Yücel & Rızvanoğlu, 2019).

Yücel and Rızvanoğlu (2019) present a qualitative study with sixteen students aged from 11 to 14 years that utilized CodeCombat with the goal of studying gender stereotypes and expectations that may impact students' behaviors and attitudes during game play. The authors conclude that CodeCombat did not engage all the participants, although it has great potential as a free and open source game. Moreover, CodeCombat was considered to have a male-centered design that makes girls feel excluded. However, as the authors themselves acknowledge the sample was small and the results cannot be generalized.

Danks et al. (2019) present teachers' experience on utilizing CodeCombat either as a supplementary or as the main resource for teaching programming to students from the first to the twelfth grade. Nearly all the teachers stated that CodeCombat engages students' interest and supports them in acquiring problem solving abilities.

Since the literature on utilizing CodeCombat in the classroom was limited we extended the Google Scholar search for other text-based serious games for programming, such as CodeSpells and Colobot. This search yielded a limited number of studies as well. Alves and Letouze (2018) presented a proposal for adjusting a curriculum for using Colobot in high school education, but they did not evaluate its effectiveness. Literature on CodeSpells aimed mainly on presenting the rationale of the game and providing preliminary results for investigating student reactions on using, modifying and creating spells in the context of the game using the programming language of Java (Esper et al., 2013, 2014).

It is clear that further research on the effectiveness of text-based serious games for programming in real classroom settings is required. Moreover, there seems to be a gap in comparing the effectiveness of text-based games and text-based programming languages for introducing secondary education students to programming concepts. These gaps in the literature guided the design of our study and the formulation of its research questions presented in the Introduction.

3 Study methodology

With the aim of investigating the research questions of the study, a teaching intervention was designed and applied in the context of the Informatics course taught in lower secondary or else junior high schools (Gymnasium) in Greece. This intervention aimed to teach basic programming concepts in Python. The participants of the study formed an experimental group that was taught programming using the serious game CodeCombat, while the control group was taught programming using the IDLE programming environment for Python. Both groups were taught by their regular informatics teacher, who is also the first author of the article. The educational material was prepared based on the official curriculum for the Informatics course in lower secondary education in Greece. The intervention took place in the school lab accordingly to the school timetable. The study took place at a school in Thessaloniki (Greece) during the school year 2018–2019.

In this section, important information about the teaching intervention, the participants and data collection instruments are presented.

3.1 Teaching intervention

The *teaching intervention based on CodeCombat* included three complete lesson plans, corresponding to sections of the game. Each lesson plan was accompanied by an evaluation worksheet. The worksheets included clear instructions, brief and concise theory, and activities of increasing difficulty that are also interconnected and make reference to material taught in previous lessons.

For the implementation of the intervention, two online classes were created on the CodeCombat website. Students of each group signed up with a unique class code and followed the instructions of the worksheets while playing the game.The intervention lasted for three didactical hours (1 didactical hour equals to 45 min). The course units, the programming concepts and the game levels utilized in each didactical hour are presented in Table 1.

A *teaching intervention using the traditional teaching method* was also designed and applied to the students in the control group of this study. It relied mainly on a textbook (basic theory and exercises) and consisted of presentation of the intended programming concepts by the instructor and problem solving using IDLE, which is the typical Integrated Development Environment (IDE) for Python. As in the experimental group the intervention lasted three didactical hours, and the worksheets used covered the same programming concepts as in the experimental group.

Table 1 Course units and CodeCombat levels

Course unit	Programming concepts	Game levels
1. Basic commands in Python	Basic syntax, arguments, strings	1–9
2. While loops in Python	While loop, strings, arguments	10–16
3. Variables	Variable naming, assignment, values and strings	17–21

3.2 Participants

The participants, enrolled in the didactic intervention, were 59 students (12–13 years old) in the first grade of a lower secondary school. As already mentioned, the participants were divided into an experimental group (EG) that was taught basic programming concepts with the use of CodeCombat, and a control group (CG) which was taught with the classical teaching approach. Out of the three classes in the first grade, one was selected (at random) as the control group, while the other two formed the experimental group. The demographics of the participants are presented in Table 2.

3.3 Data collection instruments and analysis

Information on the instruments utilized for data collection is summarized in Table 3. For each instrument, its purpose, target group and main design issues are briefly presented.

Regarding the *pre-test* we must note that in some questions excerpts of code from Scratch (Maloney et al., 2010) were used, since this was the environment that students had used in the past.

Regarding the *post-survey*, we must note that the questionnaire was adopted from a similar study with primary school students that utilized the block-based serious game Run Marco (Giannakoulas & Xinogalos, 2018). This questionnaire adopted the Technology Acceptance Model (TAM) that is specifically designed for investigating stakeholders' perceptions on the *usefulness*, *ease of use*, *attitudes* and *intentions to use* a new technology (Davis, 1989; Park, 2009). However, its questions were appropriately adjusted to: the specific type of information system, which is a serious game for programming used in a school setting; and the profile of teenagers, so as to be comprehensible and correctly perceived by them. Other models are available, with the most complete one being the Unified Theory of Acceptance and Use of Technology (UTAUT) model that was based on reviewing and combining eight other models, including TAM, and was found to outperform them in assessing the potential success of a new technology in an organization (Venkatesh et al., 2003). For a school setting, however, UTAUT comprises of a large number of items (thirty-two), while some of them were not considered relevant for teenager students. On

Table 2 Demographic data from participants

Group / Class	N	Gender	Percentage (Frequency)
1. Control group (CG) / Class1	19	Male	63.2% (12)
		Female	36.8% (7)
2. Experimental group (EG) / Class 2	21	Male	42.9% (9)
		Female	57.1% (12)
3. Experimental group (EG) / Class 3	19	Male	52.6% (10)
		Female	47.4% (9)

Table 3 Data collection instruments

Instrument	Purpose	Target group	Design
Pre-survey	Investigating: ✓students' perceived level of prior knowledge on basic programming concepts ✓students' attitudes and intentions on learning programming ✓students' familiarization and preferences on digital entertainment games	All the participants: EG (N = 40) CG (N = 19)	Closed-type questions, mainly in a 5-point Likert scale Designed by the authors
Pre-test	Assessing students' actual knowledge on basic programming concepts prior the teaching intervention	48 out of the 59 participants that declared in the pre-survey that had prior knowledge on programming: EG (N = 33) CG (N = 15)	Closed and open type questions on variables and loops Designed by the authors based on the curriculum of the Informatics course for lower secondary education in Greece
Post-test	Assessing students' knowledge on basic programming concepts, after the end of the intervention using the same test for both groups	All the participants: EG (N = 40) CG (N = 19)	Closed and open type questions on variables, comments, strings and loops Designed by the authors based on the curriculum of the Informatics course for lower secondary education in Greece
Post-survey	Investigating students' perceptions and attitudes towards the use of CodeCombat for learning programming at school	EG (N = 40)	The questionnaire proposed by Giannakoulas and Xinogalos (2018) was adopted The questionnaire is based on TAM (Davis, 1989; Park, 2009)

the other hand, the questionnaire by Giannakoulas and Xinogalos (2018) had been applied in a similar setting providing useful insights for the assessed technology.

The data collected were analyzed using SPSS. For all data collection instruments descriptive statistics were used. For the questionnaire used in the pre and post survey the Cronbach's alpha value was also calculated. The Cronbach's alpha value was 0.881 for the pre-survey questionnaire and 0.811 for the post-survey, showing good internal consistency reliability. Finally, independent samples t-tests were used for checking if there is a significant difference in the performance of the experimental and the control group in the pre and post test.

4 Study results

The most important results for each data collection instrument used in the study are analyzed in the following paragraphs, using descriptive statistics.

4.1 Pre-survey

Forty-eight out of the fifty-nine participants stated that had prior programming knowledge before the teaching intervention.

Their responses regarding the *perceived knowledge on basic programming concepts* are analyzed in Table 4. It can be seen that students believe that they do not have enough knowledge about variables (T4.1) and assignment statements (T4.2). This can be explained mainly by the association of these concepts with mathematics which is usually difficult for several students. The concept of functions seems to be more familiar to them (T4.5), while the concepts of loops and conditional statements are the most familiar ones. This may be due to the fact that the aforementioned concepts had been presented in previous years with the educational programming environment of Scratch.

The results regarding students' *intentions on learning programming* are analyzed in Table 5. This part of the questionnaire was filled in by all the students. Students' attitude about writing their own programs was not very positive (T5.1). The majority of students think that learning programming is not easy (T5.2), possibly due to the fact that a lot of novices think that learning programming requires good knowledge and skills (Shuhidan et al., 2011). However, the majority of them (T5.3) state that are interested in learning programming while playing, but are divided as to whether programming will be more interesting and fun if combined with a game (T5.4). Analogous results were recorded regarding the importance of programming for students' future (T5.5).

Although students are not particularly enthusiastic on learning programming, this is not true for games. The vast majority of students (84.7%) stated that play *entertainment digital games*, while 35.6% of them stated that play games in a daily basis (Table 6). Regarding their game type preferences, more than half the students (T6.3) prefer to play adventure or action games, while strategy games also seem to be popular (T6.2).

Table 4 Students' perceived knowledge on programming concepts

How good is your knowledge on…	N	Mean	Standard Deviation	1-Not at all Percentage (Frequency)	2-Slightly Percentage (Frequency)	3-Averagely Percentage (Frequency)	4-Much Percentage (Frequency)	5-Very Much Percentage (Frequency)
T4.1 Variables	48	1.71	1.010	54.2% (26)	31.3% (15)	8.3% (4)	2.1% (1)	4.2% (2)
T4.2 Assignment Statements	48	1.94	1.099	45.8% (22)	27.1% (13)	18.8% (9)	4.2% (2)	4.2% (2)
T4.3 Conditional Statements	48	2.81	1.197	16.7% (8)	22.9% (11)	31.3% (15)	20.8% (10)	8.3% (4)
T4.4 Loops	48	2.79	1.383	20.8% (10)	29.2% (14)	14.6% (7)	20.8% (10)	14.6% (7)
T4.5 Functions/ procedures	48	2.44	1.236	25.0% (12)	33.3% (16)	25.0% (12)	6.3% (3)	10.4% (5)

Table 5 Student intentions on learning programming

What is your opinion in learning programming?	N	Mean	Standard Deviation	1-Not at all Percentage (Frequency)	2-Slightly Percentage (Frequency)	3-Averagely Percentage (Frequency)	4-Much Percentage (Frequency)	5-Very Much Percentage (Frequency)
T5.1 I want to write my own programs	59	2.92	1.317	16.9% (10)	31.3% (13)	30.5% (18)	2.1% (8)	16.9% (10)
T5.2 I think that it is easy to learn programming	59	2.80	1.141	13.6% (8)	28.8% (17)	28.8% (17)	22.0% (13)	6.8% (4)
T5.3 I want to learn programming while playing	59	3.41	1.341	13.6% (8)	8.5% (5)	28.8% (17)	22.0% (13)	27.1% (16)
T5.4 I think that games can make programming interesting	59	2.80	1.141	13.6% (8)	28.8% (17)	28.8% (17)	22.0% (13)	6.8% (4)
T5.5 I think that programming is necessary for my future	59	2.80	1.141	13.6% (8)	28.8% (17)	28.8% (17)	22.0% (13)	6.8% (4)

Table 6 Students game type preferences

What types of games do you prefer to play?	N	Mean	Standard Deviation	1-Not at all Percentage (Frequency)	2-Rare Percentage (Frequency)	3-Enough Percentage (Frequency)	4- Often Percentage (Frequency)	5-Very Often Percentage (Frequency)
T6.1 Sports	59	2.64	1.362	23.7% (14)	30.5% (18)	16.9% (10)	15.3% (9)	13.6% (8)
T6.2 Strategy	59	3.03	1.586	23.7% (14)	20.3% (12)	15.3% (9)	10.2% (6)	30.5% (18)
T6.3 Action – Adventure	59	3.49	1.490	13.6% (8)	16.9% (10)	15.3% (9)	15.3% (9)	39.0% (23)
T6.4 Knowledge	59	2.19	1.137	32.2% (19)	35.6% (21)	18.6% (11)	8.5% (5)	5.1% (3)
T6.5 Competing (car racing)	59	2.42	1.303	30.5% (18)	25.4% (15)	27.1% (16)	5.1% (3)	11.9% (7)
T6.6 Other ……	23	3.87	1.425	7.5% (3)	2.5% (1)	7.5% (3)	21.7% (5)	47.8% (11)

Table 7 Cumulative results for the concept of "variable"

Questions	Group (N)	Correct Answer Percentage (Frequency)	Wrong Answer Percentage (Frequency)
T7.1 Does the value of a variable change during program execution?	CG (15)	46.7% (7)	53.3% (8)
	EG (33)	36.4% (12)	63.6% (21)
T7.2 Can a variable remember three values?	CG (15)	60.0% (9)	40.0% (6)
	EG (33)	63.6% (21)	36.4% (12)
T7.3 Can two variables have the same name?	CG (15)	60.0% (9)	40.0% (6)
	EG (33)	57.6% (19)	42.4% (14)
T7.4 Can a variable change its name?	CG (15)	53.3% (8)	46.7% (7)
	EG (33)	51.5% (17)	48.5% (16)

4.2 Pre-test

The results regarding *students' prior knowledge on variables* are presented in Table 7 both for the control group (CG) and the experimental group (EG). The majority of students on both groups hold insufficient knowledge on the semantics of variables (T7.1). Specifically, the majority of students answered that the value of a variable does not change during the execution of a program. However, more students answered correctly the questions that refer to the variable name (T7.3) and the values that a variable remembers (T7.2).

The analysis of the pre-test results, showed that the overall performance on the concept of "variable" for the control group (Mean = 2.2, Std.Dev. = 1.935) and the experimental group (Mead = 2.12, Std.Dev. = 1.763) did not show a statistically significant difference ($t = -0.139$, $df = 46$, $p > 0.05$).

In Table 8, the results of the questions on *"loops with a predefined number of iterations"* and *"loops with an undefined number of iterations"* are presented. "Loops with an undefined number of iterations" seem to be less difficult for students (T8.2) than "loops with a predefined number of iterations" (T8.1). The difference may be due to the fact that the "loop with an undefined number of iterations" is

Table 8 Cumulative results for the concept of "loops"

Questions on "loops"	Group (N)	Correct Answer Percentage (Frequency)	Wrong Answer Percentage (Frequency)
T8.1 "Loops with a predefined number of iterations"	CG (15)	20.0% (3)	80.0% (12)
	EG (33)	21.2% (7)	78.8% (26)
T8.2 "Loops with an undefined number of iterations"	CG (15)	33.3% (5)	66.7% (10)
	EG (33)	39.4% (13)	60.6% (20)

actually used in every program implemented with the environment of Scratch utilized in primary schools. This is a clear indication that the emphasis put by a programming environment or game on the type of programming constructs commonly used in problems solved in its context (such as "loops with an undefined number of iterations" in Scratch), favors a better understanding of the specific constructs.

The analysis of the pre-test results regarding the performance on the concept of "loops" showed that the difference between the control group (Mean = 0.53, Std. Dev = 0.834) and the experimental group (Mean = 0.61, Std.Dev = 0.827) was not statistically significant ($t = 0.282$, $df = 46$, $p > 0.05$).

4.3 Post-test

Table 9 presents the results for the control group (CG) and the experimental group (EG) on the concept of *variable*. After the teaching intervention with the traditional teaching approach (CG), the percentage of correct answers for the questions on variables ranges from 68.4% (T9.4) to 89.5% (T9.3). For the group of students that used CodeCombat (EG) the percentage of correct answers for the questions on variables ranges from 77.5% (T9.4) to 95% (T9.1).

Students' performance after the completion of the teaching intervention using CodeCombat was unexpectedly improved. For example, in the question regarding the ability to change the value of a variable (T9.1), the percentage of students in the EG that answered correctly increased from 36.4% (T7.1) before the intervention to 95% (T9.1) after the intervention. For the control group, the percentage for the same question increased from 46.7% (T7.1) before the intervention to 73.7% (T9.1) after the intervention. However, we must note that although the overall performance of the experimental group (Mean = 3.5, Std.Dev. = 1.109) on the concept of "variable" after the teaching intervention was higher than that of the control group (Mean = 3.21, Std.Dev. = 1.357), the difference was not statistically significant ($t = 0.871$, $df = 57$, $p > 0.05$).

Table 9 Cumulative results for the concept of "variable"

Questions	Group (N)	Correct Answer Percentage (Frequency)	Wrong Answer Percentage (Frequency)
T9.1 Does the value of a variable change during program execution?	CG (19)	73.7% (14)	26.3% (5)
	EG (40)	95.0% (38)	5.0% (2)
T9.2 Can a variable remember three values?	CG (19)	73.7% (14)	26.3% (5)
	EG (40)	87.5% (35)	12.5% (5)
T9.3 Can two variables have the same name?	CG (19)	89.5% (17)	10.5% (2)
	EG (40)	90.0% (36)	10.0% (4)
T9.4 Can a variable change its name?	CG (19)	68.4% (13)	31.6% (6)
	EG (40)	77.5% (31)	22.5% (9)

Table 10 Cumulative results for the concept of "loops"

Questions on "loops"	Group (N)	Correct Answer Percentage (Frequency)	Wrong Answer Percentage (Frequency)
T10.1 "Loops with an undefined number of iterations"	CG (19)	57.9% (11)	42.1% (8)
	EG (40)	77.5% (31)	22.5% (9)
T10.2 Condition of a "loop with an undefined number of iterations"	CG (19)	52.6% (10)	47.4% (9)
	EG (40)	65.0% (26)	35.0% (14)

Table 10 presents the post-test results for the concept of *loops*. Although the concept of "loops with an undefined number of iterations" appeared to be very difficult for students in the control group before the intervention (T8.2: only 33.3% of students responded correctly), their performance improved after that (T10.1: 57.9% of students responded correctly). For the experimental group the improvement was even bigger, since the percentage of correct answers increased from 39.4% (T8.2) to 77.5% (T10.1) after the intervention. The percentage of correct answers that refer to the condition of a "loop with an undefined number of iterations" (T10.2) cause difficulties to nearly half the students in the CG (47.4%) and one out of three students in the EG (35%). Although, the overall performance for the concept of "loops" for the EG (Mean = 1.43, Std.Dev. = 0.844) was better than that of the CG (Mean = 1.11, Std.Dev. = 0.994), no statistically significant difference was observed among the two groups ($t = 1.210$, $df = 30.768$, $p > 0.05$).

Table 11 presents the pot-test results for the concept of *strings and comments*. In all the questions, the majority of students in both groups answered correctly. It is clear from the results that the EG understood to a larger extent the usage of strings and comments (Mean = 5.00, St.Dev. = 1.769) than the students of the CG (Mean = 4.74, St.Dev. = 2.023), but this difference was not statistically significant ($t = 0.510$, $df = 57$, $p > 0.05$).

Table 11 Cumulative results for the concepts of "strings and comments"

Questions	Group (N)	Correct Answer Percentage (Frequency)	Wrong Answer Percentage (Frequency)
T11.1 Assigning strings	CG(19) (N = 19)	73.7% (14)	26.3% (5)
	EG (40)	77.5% (31)	22.5% (9)
T11.2 Displaying strings	CG (19)	57.9% (11)	42.1% (8)
	EG (40)	80.0% (32)	20.0% (8)
T11.3 Usefulness of comments	CG (19)	89.5% (17)	10.5% (2)
	EG (40)	97.5% (39)	2.5% (1)
T11.4 A comment starts with #	CG (19)	84.2% (16)	15.8% (3)
	EG (40)	92.5% (37)	7.5% (3)
T11.5 Single line comments	CG (19)	78.9% (15)	21.1% (4)
	EG (40)	65.0% (26)	35.0% (14)
T11.6 Python Interpreter ignores	CG (19)	73.7% (14)	26.3% (5)
Comments	EG (40)	80.0% (32)	20.0% (8)

4.4 Post-survey

Table 12 presents the results on the *perceived ease of use* of CodeCombat. The majority of students understood the goals and rules of the game very quickly and easily (Q1). It is noteworthy that most of the students started playing the game before even reading the worksheet instructions. Moreover, students agree that rarely sought the help of the teacher when completing a level (Q2: 30% agree and 35% strongly agree). The majority of students also stated that it was quite easy to detect and correct errors (Q3: 25% agree and 37.5% strongly agree). Understanding how different commands work was easy for students (Q4), while the majority of them utilized the game's instructions/hints for completing a task when being confused (Q5). Specifically, students agree (35%) or strongly agree (42.5%) that are satisfied with the tips and instructions of the game at each level. Similarly, students stated that find it easy to use CodeCombat (Q6: 30% agree and 45% strongly agree).

Table 13 presents the results on the *attitude towards the use of CodeCombat*. The vast majority of students (Q7: 75%) consider that learning programming with the use of CodeCombat is a good idea. In addition, the majority of students (Q8: 72.5%) is positive on learning programming through a game in the relevant school course.

Table 14 presents the results on the *perceived usefulness* of CodeCombat. The majority of students (Q9: 75%) declared that the lesson became more entertaining with the use of CodeCombat, while 15% of them did not seem to have much fun with the game. Also, the majority of students (Q10: 62.5%) felt that the game made it easier for them to participate in the lesson and more than half the students (Q11: 65%) consider that CodeCombat increased their interest in computer programming. Finally, students (Q12: 62.5%) believe that the game helped them comprehend the semantics of the underlying programming concepts.

Table 15 presents students' *behavioural intention* to use CodeCombat. Students seem to have different views. Although the game environment was quite attractive and students' interest remained high throughout the intervention, students were divided as to whether they would like to play with CodeCombat again (Q13). Specifically, 27% of them were neutral, while the rest of the students expressed opposite intentions regarding the future use of the game (Q13: 37.5% were negative and 35% positive). However, the students' strong positive view of educational games in general is confirmed as the majority of them (Q14: 72.5%) stated that if there were other games like CodeCombat they would like to use them.

5 Discussion

Learning programming is a challenging task as recorded in related work (Brusilovsky et al., 1997; Xinogalos, 2016). This was also reflected in students' responses recorded in the *pre-survey* used in this study. Specifically, just 28.8% of the participants consider that learning programming is easy, while the same percentage of students consider that learning programming is important for their future. On the other hand, 84.7% of the students devote time on playing digital games, even in a daily basis. This amplifies the potential of using serious games as a means of introducing novices to programming in

Table 12 Perceived ease of use for CodeCombat

Question (Giannakoulas & Xinogalos, 2018; p.2045)	N	Mean	Standard Deviation	1-Strongly Disagree Percentage (Frequency)	2-Disagree Percentage (Frequency)	3-Not sure Percentage (Frequency)	4-Agree Percentage (Frequency)	5-Strongly agree Percentage (Frequency)
Q1. "Understanding the rules of the game was easy"	40	4.00	1.013	0% (0)	7.5% (3)	27.5% (11)	22.5% (9)	42.5% (17)
Q2. "I rarely asked the teacher to help me during the game"	40	3.88	1.067	2.5% (1)	7.5% (3)	25.0% (10)	30.0% (12)	35.0% (14)
Q3. "Detecting and correcting mistakes in my code was easy"	40	3.88	1.067	0% (0)	12.5% (5)	25.0% (10)	25.0% (10)	37.5% (15)
Q4. "Understanding the functionality of the games' commands was easy for me"	40	3.95	1.300	10.0% (4)	5.0% (2)	10.0% (4)	30.0% (12)	45.0% (18)
Q5. "I often received instructions from the game in order to complete an exercise, when I was confused"	40	4.15	0.893	0% (0)	5.0% (2)	17.5% (7)	35.0% (14)	42.5% (17)
Q6. "CodeCombat is easy to use"	40	4.03	1.074	0% (0)	15.0% (6)	10.0% (4)	32.5% (13)	42.5% (17)

Table 13 Attitude towards the use of CodeCombat

Question (Giannakoulas & Xinogalos, 2018; p.2046)	N	Mean	Standard Deviation	1-Strongly disagree Percentage (Frequency)	2-Disagree Percentage (Frequency)	3-Not sure Percentage (Frequency)	4-Agree Percentage (Frequency)	5-Strongly agree Percentage (Frequency)
Q7. "I find that using CodeCombat to learn programming is a good idea"	40	4.00	1.013	0% (0)	12.5% (5)	12.5% (5)	37.5% (15)	37.5% (15)
Q8. "I'd like to have a game for learning programming in the ICT course"	40	3.83	1.152	2.5% (1)	17.5% (7)	7.5% (3)	40.0% (16)	32.5% (13)

Table 14 Perceived usefulness of CodeCombat

Question (Giannakoulas & Xinogalos, 2018; p.2047)	N	Mean	Standard Deviation	1-Strongly disagree Percentage (Frequency)	2-Disagree Percentage (Frequency)	3-Not sure Percentage (Frequency)	4-Agree Percentage (Frequency)	5-Strongly agree Percentage (Frequency)
Q9. "I believe that he lesson is more entertaining with CodeCombat"	40	3.80	1.067	7.5% (3)	2.5% (1)	15.0% (6)	52.5% (21)	22.5% (9)
Q10. "I think CodeCombat gave the opportunity to participate in the lesson more easily"	40	3.65	1.167	7.5% (3)	7.5% (3)	22.5% (9)	37.5% (15)	25.0% (10)
Q11. "I think CodeCombat has increased my interest in computer programming"	40	3.80	1.181	5.0% (2)	10.0% (4)	20.0% (8)	30.0% (12)	35.0% (14)
Q12. "I think CodeCombat helped me to understand the different ways that a program's commands can be executed"	40	3.53	1.132	10.0% (4)	5.0% (2)	22.5% (9)	47.5% (19)	15.0% (6)

Table 15 Behavioural intention to use CodeCombat

Question (Giannakoulas & Xinogalos, 2018; p.2048)	N	Mean	Standard Deviation	1-Strongly disagree Percentage (Frequency)	2-Disagree Percentage (Frequency)	3-Not sure Percentage (Frequency)	4-Agree Percentage (Frequency)	5-Strongly agree Percentage (Frequency)
Q13. "I would like to play again with the game CodeCombat in the future"	40	3.00	1.261	12.5% (5)	25.0% (10)	27.5% (11)	20.0% (8)	15.0% (6)
Q14. "If there are other similar games to learn programming, I would like to use them."	40	3.93	1.141	5.0% (2)	7.5% (3)	15.0% (6)	35.0% (14)	37.5% (15)

order to: deal with the underlying difficulties; motivate them in learning programming; and also change their attitudes and perceptions on both the importance of programming and the fact that it is not as difficult to learn as expected (Shuhidan et al., 2011).

In this study the participants were randomly separated in two groups, namely the experimental group that utilized the educational game CodeCombat and the control group that used the IDLE IDE for Python for their introduction to programming. This would give us the chance to investigate the effects of CodeCombat on learning programming, but would also help us draw some preliminary results on the added value of using a serious game in comparison with the classic approach to teaching programming (Papadakis & Kalogiannakis, 2018).

Taking into account the post-test cumulative performance of both groups on the concept of *variable*, it is concluded that the experimental group performed better than the control group. Both groups were taught the concept of variable as a memory location. However, in CodeCombat this memory location was represented as the enemy in the game in which they stored the appropriate value (enemy name) each time. This representation appeared to help students understand both the meaning of the variable name and the distinction of the name from its value. Figure 2 shows the percentage of correct answers for both groups and for each question related to the concept of "variable". Although, the experimental group showed a better comprehension of variables, the difference in the performance was not found to be statistically significant.

The post-test performance of the experimental group was better in comparison with that of the control group regarding the *loops* with an undefined number of iterations as well (Fig. 3). Students in the experimental group seemed to have a better understanding of the concept of loops and were able to observe the changes in variables' values during execution. The performance of the students in the experimental group (65%) was also better than that of the control group (52.6%) in the question regarding the correct formulation of a condition in a while loop. We must note, however, that the difference in the performance of the two groups was not statistically significant.

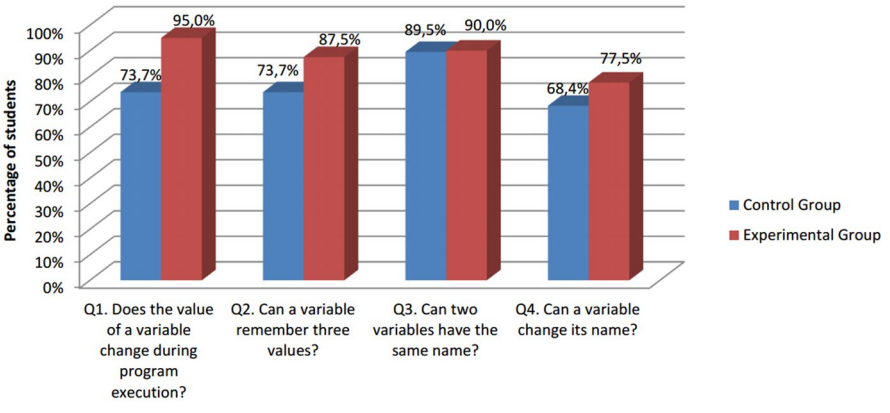

Fig. 2 Students' correct answers on questions about "variables"

This study confirmed the results of previous studies that showed that serious games (Giannakoulas & Xinogalos, 2018) or even game development approaches through the use of block-based educational environments (Papadakis, 2020) supports students in comprehending fundamental programming concepts. Moreover, it confirmed the results of a similar study that resulted in better student performance when teaching programming concepts through the game ClassCraft in comparison with the typical teaching approach, but the difference was not found to be statistically significant (Papadakis & Kalogiannakis, 2018).

The main contribution of this study lies in the fact that it investigated the effectiveness of a text-based game for programming under a real-world classroom setting. As mentioned in Sect. 2, the majority of educational games are block-based (Eguíluz et al. 2018; Giannakoulas & Xinogalos, 2020; Vahldick et al., 2014) and so is the research carried out. Research on text-based games is at least sparse and it focuses on the underlying curriculum (Alves & Letouze, 2018) or students' reactions on using a text-based programming game (Esper et al., 2013, 2014).

Although more research is necessary in order to investigate the effectiveness of CodeCombat and text-based serious games for programming in general, students' perceptions were rather positive. Specifically, positive results were recorded regarding both the *perceived ease of use* and the *attitude* towards utilizing CodeCombat for learning programming, which denotes that this is a serious game that is appropriate for novices. The results were also positive in terms of the *perceived usefulness* of CodeCombat, denoting that it makes the lesson more entertaining, it gives the opportunity to participate more easily and even more importantly it has the potential to increase students' interest in computer programming. As far as the *behavioural intention* to use CodeCombat is concerned, some interesting results were recorded. Although students evaluated positively CodeCombat, they were divided as to whether they would like to play with CodeCombat again, while they were positive in using other similar games to learn programming. This can be attributed to the following reasons: students did not have the chance to carry out more missions in

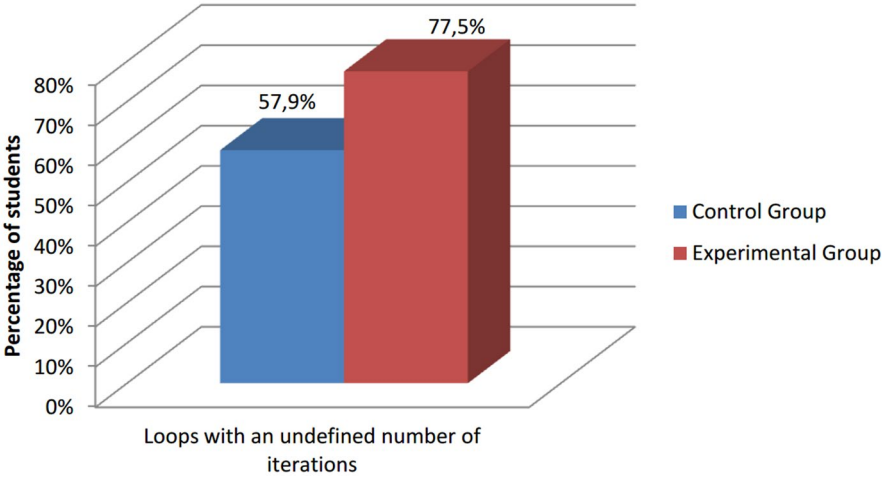

Fig. 3 Students' correct answers on "loops"

CodeCombat; the game play and 3D effects of entertainment digital games are far more immersive than that of educational games (Giannakoulas & Xinogalos, 2018); students are used to moving from the one game to the other as long as they consider that they have reached their goal in a specific game. It is clear that further research is needed in order to shed light on the issue.

6 Conclusions

Teaching and learning programming concepts is accompanied with various difficulties. Some of these difficulties are intrinsic to programming, but other difficulties are attributed to the adoption of the typical teaching approach (Brusilovsky et al. 1997; Gomes & Mendes, 2007; Papadakis & Kalogiannakis, 2018). Serious games, or else educational games, are a promising approach for introducing novices to programming concepts (Giannakoulas & Xinoglaso, 2018; Papadakis & Kalogiannakis, 2018). Several serious games for programming are available with the majority of them using a block-based notation (Eguíluz et al. 2018; Giannakoulas & Xinogalos, 2020; Vahldick et al., 2014) instead of a text- based real programming language (Alves & Letouze, 2018; Esper et al., 2013, 2014).

Empirical studies on the effectiveness of serious games for programming in the classroom are limited in comparison with the number of available games (Giannakoulas & Xinogalos, 2018). Moreover, relevant studies are based mainly on surveys (Miljanovic & Bradbury, 2018); while in most cases they are carried out by the development team of each game. This may be attributed to the challenges of utilizing a game in the classroom, namely: teachers' doubts about the actual benefits of using games in classroom (Giannakoulas et al., 2021); mapping the curriculum of a programming course to the contents of a game (Papadakis & Kalogiannakis, 2018); lack of accompanying educational material for supporting the teaching and learning of programming through a game (Giannakoulas et al., 2021).

The present study was carried out under a real-world classroom setting and aimed to contribute to the limited research on the educational effectiveness of text-based serious games that utilize conventional programming languages. More specifically, this study aimed to: investigate the educational effectiveness of the text-based serious game CodeCombat that uses Python as its programming language, as well as its acceptance by lower secondary students; compare the learning outcomes of CodeCombat with those of the typical teaching approach. Although, the study had some limitations, namely a small number of participants (59 lower secondary students) and a limited duration (3 h), some interesting results were recorded.

The study showed that CodeCombat can be successfully utilized for teaching programming in the classroom. The results obtained from the pre test and post test showed that students improved their knowledge on basic programming concepts, such as variables and loops (RQ1). Moreover, the results of a questionnaire based on TAM showed that the use of serious games as a teaching tool is readily accepted by the majority of students (RQ3). In particular, students find that playing digital games for learning programming in the classroom is a good idea and easy to use, even in the case of a text-based game that utilizes a conventional programming language.

Programming in the context of an attractive and playful environment helps students better comprehend the basic concepts of programming, but also the value of programming in general. Although their views on whether they would like to use the same game again vary, they are nonetheless willing to participate in a programming learning process using other games.

Comparing the results obtained for the two groups of students, it was observed that the performance of the students that were taught programming concepts with the use of the game was better than that of the students that were taught with the typical teaching approach (RQ2). However, this difference in the performance of the two groups was not statistically significant.

It is clear that more research is required in order to evaluate the true value of serious games as tools for teaching and learning programming. It is also important to investigate what type of serious game (block-based or text-based), or even game-based learning approach (Papadakis, 2020) should better be applied for the field of programming based on the level of education. The current study provided indications that a text-based game can be used effectively in lower secondary education, but further research has to be carried out.

References

Alves, J. R., & Letouze, P. (2018). The Curriculum Integration of a course of 'Introduction to Programming Logic'with a Serious Game-Colobot. *International Journal of Social Science and Humanity, 8*(11), 275–280.

Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., & Miller, P. (1997). Mini-languages: A way to learn programming principles. *Education and Information Technologies, 2*(1), 65–83.

Combéfis, S., Beresnevičius, G., & Dagienė, V. (2016). Learning programming through games and contests: Overview, characterisation and discussion. *Olympiads in Informatics, 10*(1), 39–60. https://doi.org/10.15388/ioi.2016.03

Danks, S., Fraumeni, B., Smrekar, M. (2019). CodeCombat: Implementation Study. Summary Report, February 2019, McREL International. Available from: https://codecombat.com/images/pages/impact/pdf/CodeCombat_ImplementationStudy_Summary.pdf

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*, 319–339.

Eguíluz, A., Garaizar, P., & Guenaga, M. (2018). An Evaluation of Open Digital Gaming Platforms for Developing Computational Thinking Skills. In *Simulation and Gaming*. InTech. https://doi.org/10.5772/intechopen.71339

Esper, S., Foster, S. R., & Griswold, W. G. (2013). CodeSpells: embodying the metaphor of wizardry for programming. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, 249–254.

Esper, S., Foster, S. R., Griswold, W. G., Herrera, C., & Snyder, W. (2014). CodeSpells: bridging educational language features with industry-standard languages. In *Proceedings of the 14th Koli calling international conference on computing education research*, 5–14.

Giannakoulas, A., & Xinogalos, S. (2020). A Review of Educational Games for Teaching Programming to Primary School Students. In Kalogiannakis, M., & Papadakis, S. (Ed.), *Handbook of Research on Tools for Teaching Computational Thinking in P-12 Education* (pp. 1–30). IGI Global. https://doi.org/10.4018/978-1-7998-4576-8.ch001.

Giannakoulas, A., Terzopoulos, G., Xinogalos, S., Satrtatzemi, M. (2021). A Proposal for an Educational Game Platform for Teaching Programming to Primary School Students. In: A. Reis et al. (eds) Technology and Innovation in Learning, Teaching and Education. TECH-EDU 2020. Communications in Computer and Information Science, vol 1384. Springer, Cham.Springer Nature Switzerland AG.

Giannakoulas, A., & Xinogalos, S. (2018). A pilot study on the effectiveness and acceptance of an educational game for teaching programming concepts to primary school students. *Education and Information Technologies, 23*, 2029–2052. https://doi.org/10.1007/s10639-018-9702-x

Gomes, A., & Mendes, A. J. (2007). An environment to improve programming education. In *Proceedings of the 2007 international conference on Computer systems and technologies* (pp. 1–6).

Kinzie, M. B., & Joseph, D. R. (2008). Gender differences in game activity preferences of middle school children: implications for educational game design. *Educational Technology Research and Development, 56*(5–6), 643–663. https://doi.org/10.1007/s11423-007-9076-z

Laporte, L., & Zaman, B. (2016). Informing Content-driven Design of Computer Programming Games: a Problems Analysis and a Game Review. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction* (p. 10). ACM. https://doi.org/10.1145/2971485.2971499

Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2014a). Designing Educational Games for Computer Programming: A Holistic Framework. *Electronic Journal of e-Learning, 12*(3), 281–298.

Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2014b). Educational games for teaching computer programming. In *Research on e-learning and ICT in Education: Technological, Pedagogical and Instructional Perspectives* (pp. 87–98). Springer.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., Eastmond, E. (2010). The Scratch Programming Language and Environment, *Transactions on Computing Education*, *10*, 4, Article 16 (November 2010), 15 pages. https://doi.org/10.1145/1868358.1868363

Maragkos K., & Grigoriadou M. (2005). The dynamics of electronic games in the learning process. A suggestion to address learning difficulties in scheduling tables. *3rd Pan-Hellenic ICT Teachers' Conference "Exploiting Information and Communication Technologies in Teaching Practice"*, Syros, 13–15 May 2005.

Michael, D. R., & Chen, S. L. (2005). *Serious games: Games that educate, train, and inform*. Muska & Lipman/Premier-Trade.

Miljanovic, M.A., & Bradbury, J.S. (2018). A Review of Serious Games for Programming. In: Göbel S. et al. (eds) Serious Games. JCSG 2018. Lecture Notes in Computer Science, vol 11243, 2018, Springer, Cham, https://doi.org/10.1007/978-3-030-02762-9_21

Mitchell, A., & Savill-Smith, C. (2004). *The use of computer and video games for learning. A review of the literature*. Learning and Skills Development Agency.

Mladenović, M., Boljat, I., & Žanko, Ž. (2018). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies, 23*, 1483–1500. https://doi.org/10.1007/s10639-017-9673-3

Nazry, N., Nazrina, M., & Romano, D. M. (2017). Mood and learning in navigation-based serious games. *Computers in Human Behavior, 73*, 596–604.

Papadakis S., Kalogiannakis M. (2018) Using Gamification for Supporting an Introductory Programming Course. The Case of ClassCraft in a Secondary Education Classroom. In: Brooks A., Brooks E., Vidakis N. (eds) Interactivity, Game Creation, Design, Learning, and Innovation. ArtsIT 2017, DLI 2017. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 229. Springer, Cham. https://doi.org/10.1007/978-3-319-76908-0_35

Papadakis, S. (2020). Evaluating a game-development approach to teach introductory programming concepts in secondary education. *International Journal of Technology Enhanced Learning, 12*(2), 127–145. https://doi.org/10.1504/ijtel.2020.106282.

Park, S. Y. (2009). An analysis of the technology acceptance model in Understanding University students' behavioral intention to use e-learning. *Educational Technology & Society, 12*(3), 150–162.

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

Shuhidan, S. M., Hamilton, M., & D'Souza, D. (2011). Understanding novice programmer difficulties via guided learning. ACM, *ITiCSE* '11, p. 213–217.

Siegle, D. (2017). Technology: Encouraging creativity and problem solving through coding. *Gifted Child Today, 40*(2), 117–123.

Vahldick, A., Mendes, A. J., & Marcelino, M. J. (2014). A review of games designed to improve introductory computer programming competencies. In *Frontiers in Education Conference* (FIE) (pp. 1–7). IEEE. https://doi.org/10.1109/FIE.2014.7044114

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: Toward a unified view. *MIS quarterly*, 425–478.

Virvou, M., Katsionis, G., & Manos, K. (2005). Combining Software Games with Education: Evaluation of its Educational Effectiveness. *Educational Technology & Society, Journal of International Forum of Educational Technology & Society and IEEE Learning Technology Task Force, 8*(2), 54–65.

Weintrop, D., and Wilensky, U. (2017). Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms. *ACM Trans. Comput. Educ. 18*, 1, Article 3 (October 2017), 25 pages. https://doi.org/10.1145/3089799

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies, 21*(3), 559–588.

Yücel, Y., & Rızvanoğlu, K. (2019). Battling gender stereotypes: A user study of a code-learning game, "Code Combat", with middle school children. *Computers in Human Behavior, 99*, 352–365.

Authors and Affiliations

Chrysoula Kroustalli[1] · Stelios Xinogalos[2]

Chrysoula Kroustalli
kroustac@gmail.com

[1] Master of Science in Sciences of Education and Lifelong Learning, University of Macedonia, Thessaloniki, Greece

[2] Department of Applied Informatics, School of Information Sciences, University of Macedonia, 156 Egnatia Street, 54636 Thessaloniki, Greece