A pilot study on the effectiveness and acceptance of an educational game for teaching programming concepts to primary school students

Abstract

Educational games are increasingly used in informal and formal educational settings for promoting active learning and gaining students' interest in cognitively demanding subjects, such as programming. However, empirical studies that investigate the true impact of educational games on teaching and learning programming, especially to small aged students, are limited. This article presents the results of a pilot study that utilized the educational game *Run Marco* for teaching basic programming concepts to primary school students. Students' performance was studied through specially designed worksheets, while their acceptance of the intervention was evaluated through a questionnaire that was based on the principles of the Technology Acceptance Model (TAM). The results of the pilot study showed that the educational game supported students in comprehending basic programming concepts, while the results regarding the acceptance of its usage in the learning process were quite positive. However, the game did not succeed in raising students' interest as expected and further research is necessary in order to study the reasons for this fact and make informed choices on designing and utilizing such games.

Keywords: educational games, programming, primary school students, technology acceptance model (TAM)

1. Introduction

Computer programming has been introduced into many curricula of different ages in recent years as it is considered a very important skill (Fesakis & Serafeim, 2009). Teaching programming to novice programmers seems to be accompanied by several problems (Fesakis & Serafeim, 2009; Laporte & Zaman, 2016; Xinogalos, 2016) due to several factors (Piteira & Haddad, 2011). The difficulty in understanding the abstract programming concepts, as well as the traditional teaching approach that is characterized by lack of interactivity, reduces students' motivation and additionally their interest in learning programming (Piteira & Haddad, 2011). In order to overcome these difficulties different approaches (Xinogalos & Satratzemi, 2004) and types of educational programming environments (Xinogalos et al., 2017) have been proposed, such as programming microworlds (Xinogalos et al., 2006), flowchart-based programming environments (Xinogalos, 2013) and more recently educational games (Kazimoglu et al., 2012; Malliarakis et al., 2014b; Piteira & Haddad, 2011; Vahldick et al., 2014). Moreover, studying the source code of existing games and extending them (Theodoraki & Xinogalos, 2014) or even building a new game based on an existing one (Xinogalos, 2018), as well as using online games in the context of

programming contests (Combéfis et al., 2016) have been proposed as alternative teaching approaches for introducing novices to programming.

Using educational games in the teaching process can bring many benefits. Educational games -among others- can create an attractive and interactive learning context that motivates students for practicing with programming, in which students try to achieve specific goals, providing them at the same time, feedback, challenge and scaffolding (Kazimoglu et al., 2012; Laporte & Zaman, 2016; Malliarakis et al., 2014b). For these reasons, several educational, or else serious games for learning programming have been developed and the first results of their evaluation as a means of learning are quite positive (Malliarakis et al., 2017). When primary school students are concerned, utilizing educational games for an introduction to programming concepts as a means of promoting Computational Thinking (CT) is considered even more important.

Several educational games have been developed that could be used for introducing young students to fundamental programming concepts and CT, such as (in alphabetical order):

- *CodeMonkey* (https://www.playcodemonkey.com)
- *Getgoding* (http://www.hepis.gr/getcoding)
- *Kodable* (https://www.kodable.com)
- *Lightbot* (http://lightbot.com)
- *LightBot 1.0* (http://armorgames.com/play/2205/light-bot)
- *Lightbot 2.0* (http://armorgames.com/play/6061/light-bot-20)
- *Program Your Robot* (Kazimoglu et al., 2012)
- *RapidRouter* (https://www.codeforlife.education/play/rapid-router)
- *RunMarco* (https://www.allcancode.com/web)

All the aforementioned games are classified as puzzle games and do not require prior programming knowledge. We consider *RunMarco* as a good choice for introducing primary school students to programming concepts for the following reasons:

- The game is translated into 26 languages and is suitable for students aged from 6 to 12 years.
- *Run Marco,* as well as *Rapid Router,* covers most of the basic programming concepts in relation to the rest of the games reported before, while in its next version it is intended to cover more concepts, such as variables, data structures, functions, logical conditions, and procedures, offering additional opportunities for teachers and students.
- In its current version the game offers a satisfactory number of 36 levels, while in its next version the game will allow the design of new levels both by teachers and students through a specially designed *level editor*.
- *Run Marco* supports *student performance monitoring* in real time. This feature is also available in *Rapid Router*, *Codemonkey* and *Kodable*.

Although several educational games for programming are freely available, empirical studies that investigate their effectiveness as tools for learning introductory computer programming refer mainly to higher education students (Malliarakis et al., 2014a; Malliarakis et al., 2017; Orehovački et al., 2015), while the studies that refer to small aged students are quite few (Gibson & Bell, 2013; Kazimoglou et al., 2012). However, as Zaharija et al. (2013) state, games can play an important role in teaching programming to young students, since the classic teaching approach is not appropriate for their age and does not help in dealing with difficulties such as the short attention span that characterizes them. Although the usage of educational games for teaching young students programming is considered important, we rarely find in the existing literature instructions or strategies for appropriately incorporating them in the learning process (Piteira & Haddad, 2011). In order to contribute in this direction, we designed a pilot teaching intervention that uses an educational game to teach basic programming concepts to young novice programmers.

The aim of the study presented in this article is to investigate the effectiveness of the educational game "Run Marco" in teaching basic programming concepts to primary school students. More specifically, we tried to investigate students' attitudes on using an educational game that teaches basic programming concepts and moreover we tried to investigate if the use of the game had positive impact on students' motivation to learn programming. For this purpose we designed and applied a didactical intervention in the context of the "Information and Communication Technologies" (ICT) course, which is integrated into the curriculum of elementary schools in Greece. In order to collect data we designed special evaluation worksheets, as well as a questionnaire in accordance with the principles of the Technology Acceptance Model (TAM) that were completed by the students.

The rest of the article is organized as follows. Section 2 presents briefly the results of studies on utilizing educational games for introducing students to programming. Section 3 presents the main characteristics of the educational game *Run Marco* that was utilized, while section 4 presents the research questions and the methodology of the study. Section 5 analyzes the results of student replies to the worksheets and section 6 analyzes the results of students' replies in the questionnaire. Finally, in section 7 the findings of both the evaluation worksheets and the questionnaire are discussed, while in section 8 conclusions are drawn and proposals for further research are presented.

2. Relevant Work

In this section the results of studies that utilized the aforementioned educational games are briefly summarized. The first two studies do not refer to primary school students that are our target group, but were included since they were the only ones we could find that utilized the well-known game of *Lightbot*.

Mathrani et al. (2016) utilized *LightBot 2.0* in the context of a diploma computing course in a non-university education provider offering ICT related subject courses at different study levels in New Zealand, in order to engage students in learning and enhance their programming skill sets. Two separate student cohorts were invited to participate in the game-based learning experiment, one not having prior programming knowledge and the other having little prior programming knowledge after having recently completed a programming course. Findings reveal that educational games add a fun element in the learning process and students considered the game as an effective way to learn programming.

In another study Lopez et al. (2016) present how *Lightbot 1.0* has been used in two universities in Peru since 2015 for introducing students to three programming concepts: abstraction, function and reuse. One university uses Java while the other one uses Python as a programming language. *Ligthbot 1.0* is used in the first laboratory session. Findings reveal students' positive attitude regarding the use of the game in the teaching process and in addition there seems to be a slight improvement in students' performance after using the game.

Kazimoglu et al. (2012) designed an educational game, named *ProgramYourRobot* where students can practice and develop their skills in CT with little or no programming knowledge. Their initial feedback from a survey response group of 25 students, who played their game as a voluntary exercise, shows that the majority of participants found the game interesting and moreover well suited to help students to understand introductory programming constructs.

Karadeniz et al. (2014) presented a case study on how 5 years old pupils, were introduced to basic programming constructs, such as sequence commands, conditions and loops, combining traditional teaching with experiential games in the classroom and the game *Kodable* as well. Findings reveal that pupils enjoyed playing with *Kodable* and moreover they had satisfactory performance on both the especially designed evaluation worksheets and the game's activities.

Kandroudi & Bratitsis (2016) presented a pilot teaching intervention to teach basic programming concepts to pupils aged 5 to 7, combining the educational game *Kodable* and the educational programming environment *ScratchJr*. Pupils were introduced into a new programming concept, initially using *Kodable* and after that, they had to create their own story in *ScratchJr* utilizing games' corresponding commands, alternating sometimes the order they used the two environments. Findings reveal that teaching basic programming concepts seemed to be easier for pupils when *Kodable* was first used and after that the programming environment *ScratchJr*.

3. The Game Run Marco

Run Marco (Figure 1) is an educational game that aims to teach the player, basic programming concepts, such as sequence commands, iteration and conditions. The game is geared for ages 6-12 and is translated into 26 languages. It is available

through any contemporary browser on its official website https://www.allcancode.com/web and in addition, there are also versions for iPad and Android. Players are asked to help "*Marco*" - the main character -, who is lost in the jungle, so he can find his friends again. For this purpose, at each level of the game, players give appropriate commands to "*Marco*", guiding him on a path. This is accomplished through a visual programming language, namely "*Blockly*", that is the standard visual programming language developed by "*Google*" and used by "*Code.org*". In particular, players "*drag and drop*" simple commands, such as "*step forward*", over a white area that is the programming editor of the game, forming a set of commands (Figure 1). When the player presses the "*run*" button "Marco" reacts by executing the commands written for him. Players can modify the code they write until they reach its level's goal. As players successfully complete the game levels, they unlock more complex commands while at the same time, the complexity of the game increases. The game rewards the player every time he completes a level giving him a maximum of three stars, depending on the efficiency of the solution he/she gave. The player has the ability to resolve a level again, by giving a more efficient solution. In order to support the player in understanding the graphical interface but also whenever a new command is introduced, the game displays to the player a series of illustrated cards that the player can see one after the other.

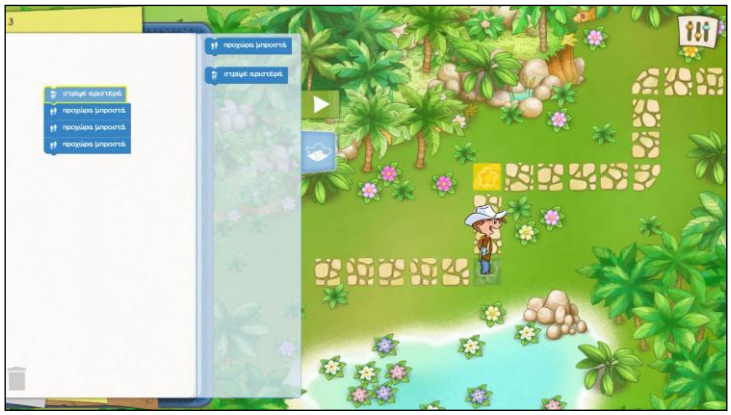

Figure 1. Screenshot from Run Marco.

The game includes 36 levels and is divided into 6 sections. Table 1 presents the game's sections and the content (goals) of each section.

Table 1. Sections and content of Run Marco

Section	Content	Game levels
1. Sequence of Commands	Sequence commands execution	1-9
2. Iteration I (Repeat Times)	• Loops with a predefined number of iterations • Nested loops	10-19
3. Iteration II (Repeat While)	Loops with undefined number of iterations	20-22
4. Conditions I	Simple if	23-30
5. Conditions II	If…else	31-33
6. Conditions III	Nested if	34-36

In its current version, the game allows the teacher to monitor students' progress in real time, while in future versions of the game there will be a new level creation function. In addition, in the future, other programming concepts such as "*variables*", "*data structures*", and "*functions*" will be included according to its designers.

4. The Study

The research presented in this article was a six step process, as shown in Figure 2. More details about the steps of the study are presented in the sections that follow.

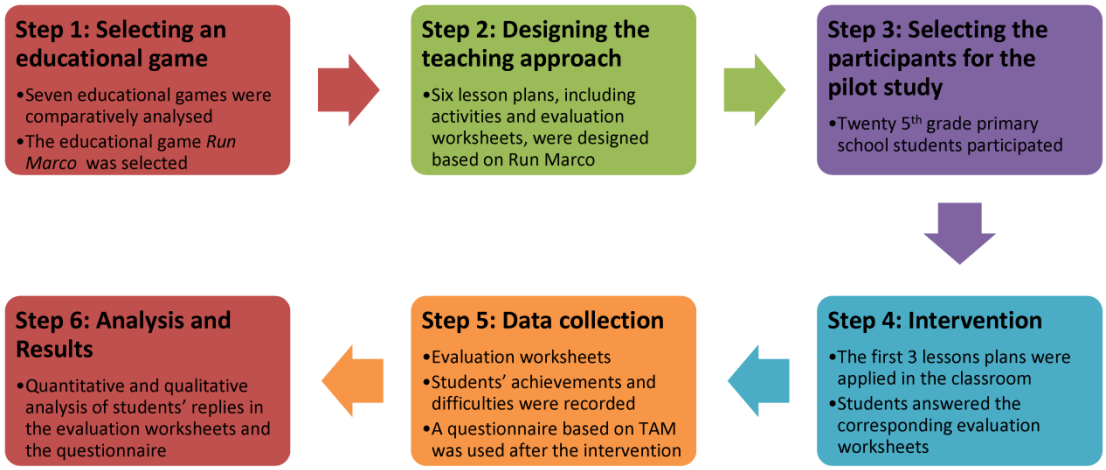

Figure 2. The research process of the pilot study.

4.1 Research Questions

The study presented in this article aimed at investigating the following research questions:

- **Research Question 1 (RQ1):** Does *Run Marco* support primary school students in comprehending programming concepts?
- **Research Question 2 (RQ2):** Do primary school students accept the utilization of educational games in learning basic programming concepts?
- **Research Question 3 (RQ3):** Did *Run Marco* cause the interest of students?

4.2. Methodology

The study presented is a pilot didactic intervention, which was conducted during the school year 2016–2017 in a primary school of Thessaloniki in Greece, within the "Information and Communication Technologies" (ICT) course. In the context of the ICT course, 5th and 6th grade students are taught basic programming concepts. The

didactic intervention took place in the school's computer lab and twenty 5th grade students participated.

First we designed a teaching proposal that includes six complete *lesson plans*, according to the "Run Marco" game sections. Each lesson plan is accompanied with an auxiliary *activities worksheet*, which provides instructions for solving each level of the game. Moreover, an *evaluation worksheet* that includes game-related exercises and in which students answer using paper and pencil has been prepared for each lesson.

Due to time restrictions, it was decided to teach the first three lessons as planned in the teaching proposal. The duration of the intervention presented was three hours and the content and goals of each hour are presented in Table 1 (rows 1-3).

At the beginning of each lesson, students were introduced to the basic programming concepts and then they were given time to play with the levels of the game that correspond to the specific concepts. The students were divided into pairs. Initially, students were given the activities sheet and after that, students had time to play with the levels of the game that corresponded to the particular lesson plan.

Students did not complete the levels of the game simultaneously. When students completed the levels of the game, they were given the corresponding *evaluation worksheet* and were asked to fill it in by themselves in class using paper and pencil. Before completing the worksheet, students received appropriate guidance from the teacher in order to understand the requirements of each exercise. If some students were not able to complete the worksheet during the lesson, they were given some extra time.

4.3 Data Collection

In order to collect data for investigating the research questions of the study the following instruments were used:

- The teacher recorded students' *achievements and difficulties during game play* (carrying out the activities incorporated in the game – *activities worksheet*).
- *Analysis of students' replies in the evaluation worksheets*. Both a quantitative and qualitative analysis is carried out.
- A specially designed *questionnaire* was distributed to students after the end of the intervention. The questionnaire (Tables 5-8) was designed in accordance with the Technology Acceptance Model (TAM) and includes questions aimed at investigating students' *perceived usefulness* and *ease of use* of the utilized serious game, as well as their *intention to use* it in the future (Davis, 1989; Lederer et al., 2000; Legris et al., 2003; Park, 2009; Alharbi and Drew, 2014).

5. Analysis of students' performance on the worksheets

In this section, the results of analyzing the activities and evaluation worksheets, as well as students' performance at the different levels of the game that was recorded by the teacher, are presented. The worksheets included open-ended exercises, closed-type exercises, and exercises in which students were asked to write a small program using the corresponding commands they had used in *Run Marco*. Students answered the evaluation worksheets on paper and pencil. The first evaluation worksheet is described in more detail in order for the reader to understand the type of activities students dealt with both during the learning and evaluation process.

5.1 Sequence commands

This first evaluation worksheet was filled by 19 of the 20 students that were present in the lesson.

Question 1.1 (Q1.1): *What is an algorithm?*

The definition of an *algorithm* was given at the beginning of the course and this open-ended question intended to investigate whether students understood that *an algorithm is a clearly defined sequence of steps that helps us solve a problem*. At the beginning of the lesson the overwhelming majority of students stated that had not heard the concept of algorithm before. In the context of the relevant question at the end of the lesson, the majority of students (78.95%) exhibited a satisfactory comprehension for the concept of algorithm.

Question 1.2 (Q1.2): *What is a computer program?*

At the beginning of the first lesson the following definition was given to students for a *computer program*: "*A computer program is an algorithm that has been written in such a way (encoded in another format) so it can be executed from a computer*". Although the majority of students did not know what a computer program is before this lesson, the majority of them (78.95%) managed to provide a satisfactory description after completing the first lesson.

Question 1.3 (Q1.3): *Guide Marco on the stone path in order to reach the yellow tile (Figure 3). You can use the commands "step forward", "turn left" and "turn right".*

In this exercise, students were asked to write the appropriate sequence of instructions that solves the puzzle. The majority of students (68.42%) managed to implement a completely correct program. However, nearly one third of the students did not manage to implement a correct program (31.58%).

Figure 3. Question 1.3 of the first evaluation worksheet.

Most of the errors were attributed to students' *misconceptions* regarding the semantics of the "turn left" and "turn right" commands. What seemed to make it difficult for students was that they could not understand that, for example, the "turn right" command results in the hero turning in the direction of his right hand, regardless of his direction on the path (Up, Down, Right, Left). On the contrary, students considered that the command "turn right" turns the hero to the right of the screen, regardless of the direction the hero has on the path. Also, some students believed that the commands that change the direction of the hero, at the same time have the effect of moving him one step ahead. These potential misconceptions should be taken into account from teachers using the game in order to avoid them from the very beginning.

Question 1.4 (Q1.4): *The following program is being set up to guide Marco onto this path to reach the yellow tile (Figure 4).*

Figure 4. Question 1.4 of the first evaluation worksheet.

1.4a. Does the program work correctly?
1.4b. If not, rewrite it, in order to work properly.

With the exception of one student (5.26%), all other students managed to *debug* the program in paper and pencil and detect the error. Fewer students, but still the majority of them (73.68%) managed to correct the program.

Question 1.5 (Q1.5): *Draw the path that Sophia will follow if she executes the commands of the program shown in Figure 5, by painting the appropriate squares.*

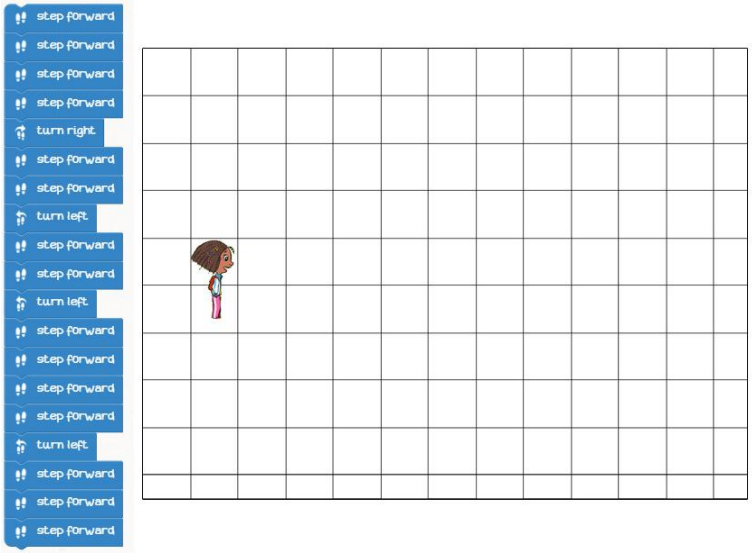

Figure 5. Question 1.5 of the first evaluation worksheet.

A very large percentage of students (78.95%) responded correctly to the fifth question.

Table 2 shows the cumulative results of student responses in the first evaluation worksheet. In almost all the questions, with the exception of question 3, at least 73% of the students answered correctly.

Table 2. Cumulative results of student responses in the first worksheet.

	Q 1.1	Q 1.2	Q 1.3	Q1.4a	Q 1.4b	Q1.5
Correct answers	78.95% (15)	78.95% (15)	68.42% (13)	94.74% (18)	73.68% (14)	78.95% (15)
Wrong answers	21.05% (4)	21.05% (4)	31.58% (6)	5.26 % (1)	26.32 % (5)	21.05% (4)
Total	100% (19)	100% (19)	100% (19)	100% (19)	100% (19)	100% (19)

In addition, based on teacher's observation, all students successfully completed the levels 1 through 9 of the game, which correspond to the first lesson.

It is clear from the results on student performance both from the activities and the evaluation worksheets that students understood to a large extent both the functionality of the game commands and the concept of serial execution of commands (sequence commands) and were able to:

- give in simple words the *definitions of algorithm and program* (Q1.1 and Q1.2)
- to *implement* a simple sequential program (Q1.3)
- to *debug* a simple sequential program (Q1.4)
- to *execute in paper and pencil* a simple sequential program (Q1.5)

5.2 Loops with a predefined number of iterations

This worksheet was filled in by 20 students in the classroom. Initially, the questions are presented and then the cumulative results are summarized and briefly analyzed.

Question 2.1 (Q2.1): *Draw the path that Sophia will follow if she executes the commands of the program shown in Figure 6, by painting the appropriate squares.*

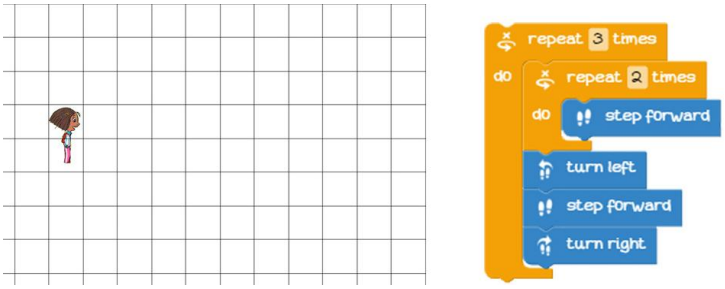

Figure 6. Question 2.1 of the second evaluation worksheet

Question 2.2 (Q2.2): *Fill in the next loop appropriately in order to guide "Marco" on the stone path to reach the yellow tile (Figure 7). You can use the commands "step forward", "turn left" and "turn right".*

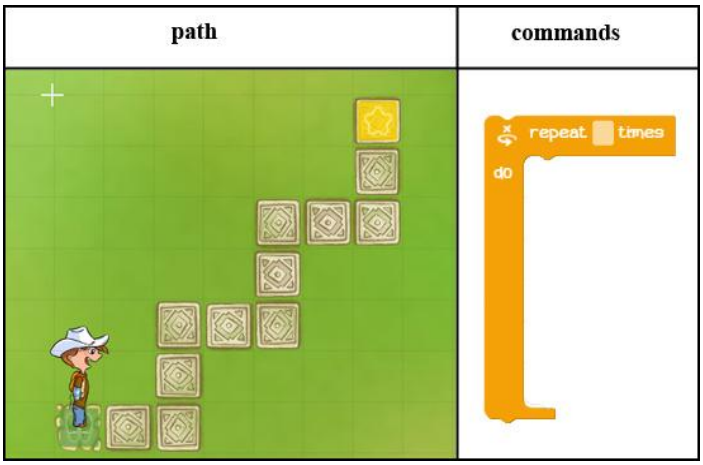

Figure 7. Question 2.2 of the second evaluation worksheet.

Question 2.3 (Q2.3): *Fill in the next loop appropriately in order to guide "Marco" on the stone path to reach the yellow tile (Figure 8).*

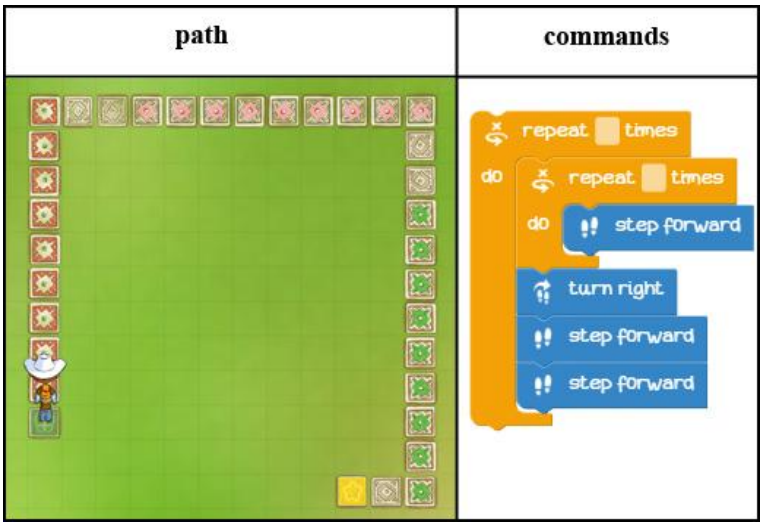

Figure 8. Question 2.3 of the second evaluation worksheet.

Table 3 presents the cumulative results of student responses in the second evaluation worksheet. In all the questions, at least 60% (Q1: 65%, Q2: 60%, Q3: 70%) of the students answered correctly. Consequently, students managed to *execute in paper and pencil* and *implement* either from scratch or incomplete programs with nested loops with a predefined number of iterations. However, nearly 10% lesser students managed to complete these activities correctly in comparison with the corresponding activities on the sequential structure.

Table 3. Cumulative results of student responses in the second worksheet.

	Q2.1	Q2.2	Q2.3
Correct answers	65.00% (13)	60.00% (12)	70.00% (14)
Wrong answers	35.0% (7)	40.00 % (8)	30.00% (6)
Total	100% (20)	100% (20)	100% (20)

In addition, based on the teacher's observation, almost all students successfully completed the levels 10 through 12 of the game, which were similar to Q2.1, as well as the game's levels 13 through 15, which were similar to Q2.2, requiring the discovery of a pattern that is repeated. On the other hand, students faced many difficulties at completing successfully the game's levels 16 through 19, concerning the use of nested loops, which were similar to Q2.3.

Based on the performance of the students on the second evaluation worksheet, but also on their performance in the game, we conclude that students understood to a good degree how the repeat command "*Repeat x times*" works. In addition, students seem to face more difficulties in understanding the concept of *nested loops*, which is quite demanding for their age.

5.3 Loops with an undefined number of iterations

This worksheet was filled by 19 of 20 students in the classroom.

Question 3.1 (Q3.1): *Draw the path that Sophia will follow, from the green tile to the yellow tile, if she executes the commands of the program shown in Figure 9, by painting the appropriate squares of the grid.*

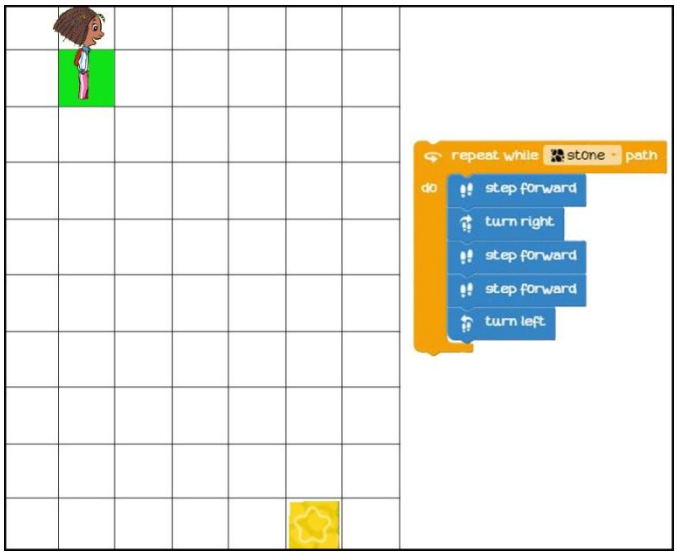

Figure 9. Question 3.1 of the third evaluation worksheet.

Question 3.2 (Q3.2): *Fill in the next loop appropriately in order to guide "Marco" on the stone path to reach the yellow tile (Figure 10). You can use the commands "step forward", "turn left" and "turn right".*

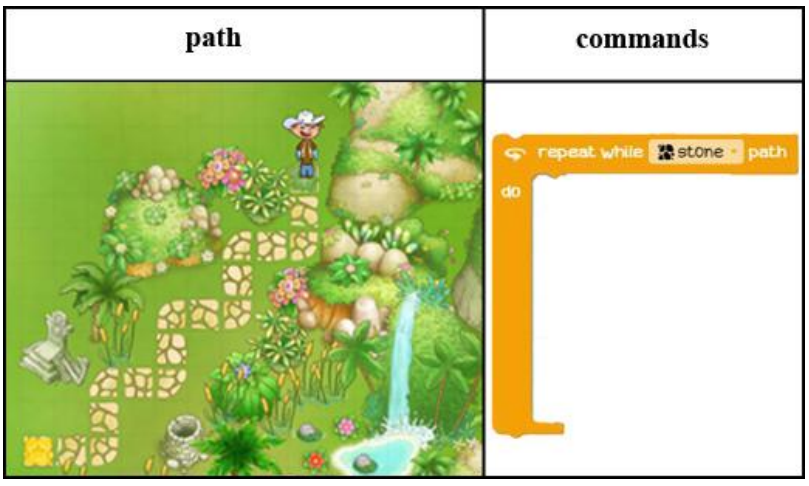

Figure 10. Question 3.2 of the third evaluation worksheet.

Question 3.3 (Q3.3): *Fill in the next loop appropriately in order to guide "Marco" on the stone path to reach the yellow tile (Figure 11). You can use the commands "step forward", "turn left" and "turn right".*

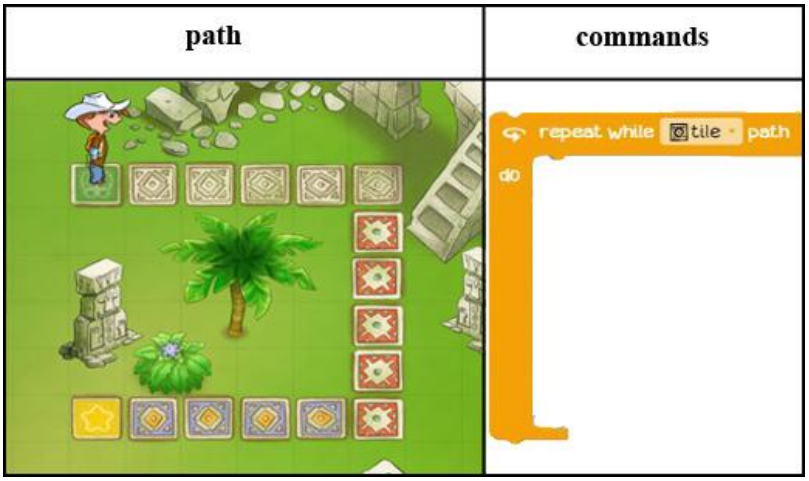

Figure 11. Question 3.3 of the third evaluation worksheet.

Question 3.4 (Q3.4): *Fill in the next loop appropriately in order to guide "Marco" on the stone path to reach the yellow tile (Figure 12). You can use the commands "step forward", "turn left" and "turn right".*

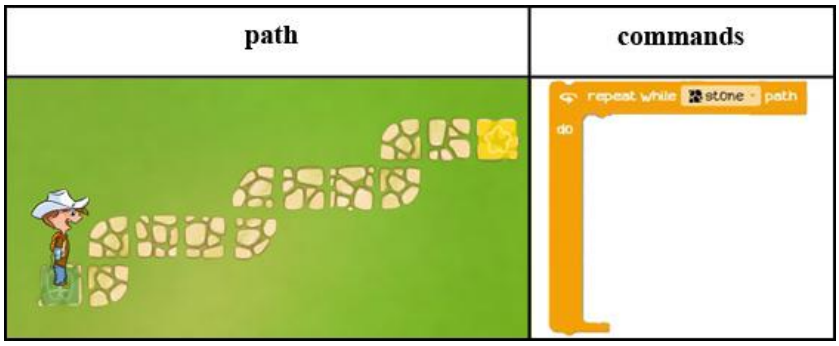

Figure 12. Question 3.4 of the third evaluation worksheet.

Table 4 shows the cumulative results of student responses in the third evaluation worksheet.

Table 4. Cumulative results of student responses in the third worksheet.

	Q3.1	Q3.2	Q3.3	Q3.4
Correct answers	84.21% (16)	52.63% (10)	68.42% (13)	42.11% (8)
Wrong answers	15.79% (4)	47.37% (9)	31.58% (6)	57.89 % (11)
Total	100% (19)	100% (19)	100% (19)	100% (19)

The vast majority of students (84.21%) responded correctly to Q3.1 that required executing a program in paper and pencil. In Q3.2 and Q3.3, which concerned the identification of a repeating pattern for finishing an incomplete program, the majority of students responded correctly. The better performance of students in Q3.3 compared to Q3.2 and Q3.4, may be due to the fact that in Q3.3 the tiles of different colours used to create the pattern made the pattern more distinct. Moreover, based on the

teacher's observation, students did not have difficulty completing the appropriate condition at game levels 21 and 22, concerning the use of the structure *"Repeat While"*.

6. Analysis of the questionnaire replies

In this section, students' replies in the questionnaire are analyzed. The questionnaire consisted only of closed-type, Likert-scale questions, where the possible answers were: 1 – Strongly Disagree, 2 – Disagree, 3 – Not Sure, 4 – Agree and 5 – Strongly Agree. In Tables 5-8 the percentage of replies falling in each one of these 5 responses of each question, as well as the mean value (in the scale of 1 to 5) and the standard deviation of students' replies is presented, while the mean values are also presented in Figures 14-17 using charts. The questionnaire was designed in accordance with the Technology Acceptance Model (TAM) that is briefly described in section 6.1.

6.1 Technology Acceptance Model (TAM)

The Technology Acceptance Model (TAM) presented in Figure 13, is a theoretical model proposed by Davis in 1989 (Davis, 1989) in order to predict or to explain why users accept or reject a new Information Technology (Legris et al., 2003). According to this model, the two most important factors that explain the adoption or rejection of a new Information Technology are *Perceived Ease of Use* and *Perceived Usefulness* (Davis, 1989).

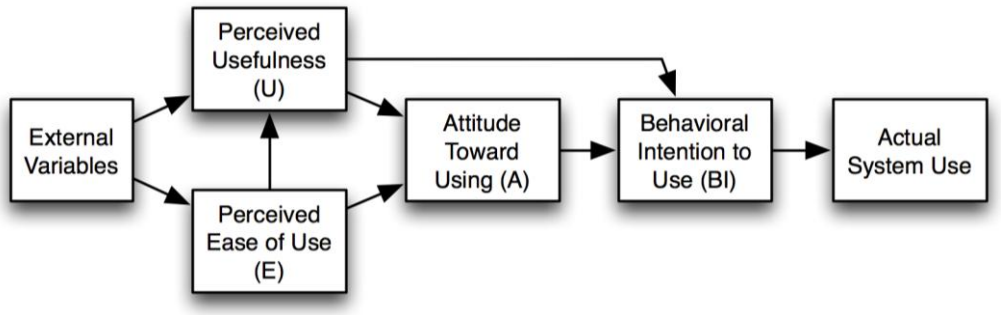

Figure 13. Technology Acceptance Model - TAM (Davis, 1989).

Davis (1989, p.320) defined Perceived Usefulness as "*the degree to which a person believes that using a particular system would enhance his or her job performance*" and Perceived Ease of Use as "*the degree to which a person believes that using a particular system would be free of effort*".

The other two factors that affect the *actual use* of a new Information Technology are *Attitude toward Use (AT)* and *Behavioral Intention to Use (BI)* (Lederer et al., 2000; Park, 2009). The *Attitude toward Use* is concerned with the user's evaluation of the

desirability of employing a particular information system application while *Behavioral Intention to Use* is the measure of the likelihood of a person employing the application (Lederer et al., 2000; Surendran, 2012).

TAM has proven to be a useful theoretical model which helps to understand and explain user behavior in an information system (Legris et al., 2003; Shih-Chi et al., 2011). It has been tested in many empirical studies by researchers worldwide, in order to understand the acceptance of different types of information systems and results have been reliable (Alharbi and Drew, 2014; Chuttur, 2009; Legris et al., 2003; Surendran, 2012).

6.2 Perceived ease of use for the Run Marco game

Table 5 presents students' answers regarding the perceived ease of use of *Run Marco*. In Figure 14 a chart with the mean value of students' replies in each question is presented.

Table 5. Perceived ease of use for *Run Marco*.

Question	1 - Strongly Disagree	2 - Disagree	3 - Not Sure	4 – Agree	5 - Strongly Agree	Mean	Standard deviation
Q1. Understanding the rules of the game was easy	0	0	0	36.84	63.16	4.63	0.50
Q2. I rarely asked the teacher to help me during the game	0	5.26	5.26	42.11	47.37	4.26	0.81
Q3. Detecting and correcting mistakes in my code was easy	0	5.26	15.79	21.05	57.89	4.32	0.95
Q4. Understanding the functionality of the games' commands was easy for me	0	0	10.53	26.32	63.16	4.53	0.70
Q5. I often received instructions from the game in order to complete an exercise, when I was confused.	10.53	21.05	15.79	36.84	15.79	3.26	1.28
Q6. "Run Marco" is easy to use	0	5.26	5.26	15.79	73.68	4.58	0.84

All the students agreed (Q1: 36.84% agree and 63.16% strongly agree) that it was easy to understand the rules of the game, confirming that the game is characterized by clear rules. Also, the vast majority of students (89.48% - Q2: 42.11% agree and 47.37% strongly agree) declared that rarely needed help from the teacher during the game, while the same proportion of them (89.48% - Q4: 26.32% agree and 63.16% strongly agree) considers that it was easy to understand the functionality of the games' commands. In addition, a large proportion of them stated that, finding and correcting bugs in their code was easy (78.95% - Q3: 21.05% agree and 57.89%

strongly agree). On the other hand, students seem to be divided in terms of the frequency of the game's auxiliary messages, as half of them (52.63%) are satisfied with this game mechanism (Q3: 36.84% agree and 15.79% strongly agree). Based on this fact, we can conclude that an improvement in the games' scaffolding mechanisms may be required, particularly in the case of more challenging activities. Finally, the vast majority of students (89.47% - Q6: 15.79% agree and 73.68% strongly agree), finds it easy to use the game *Run Marco*.

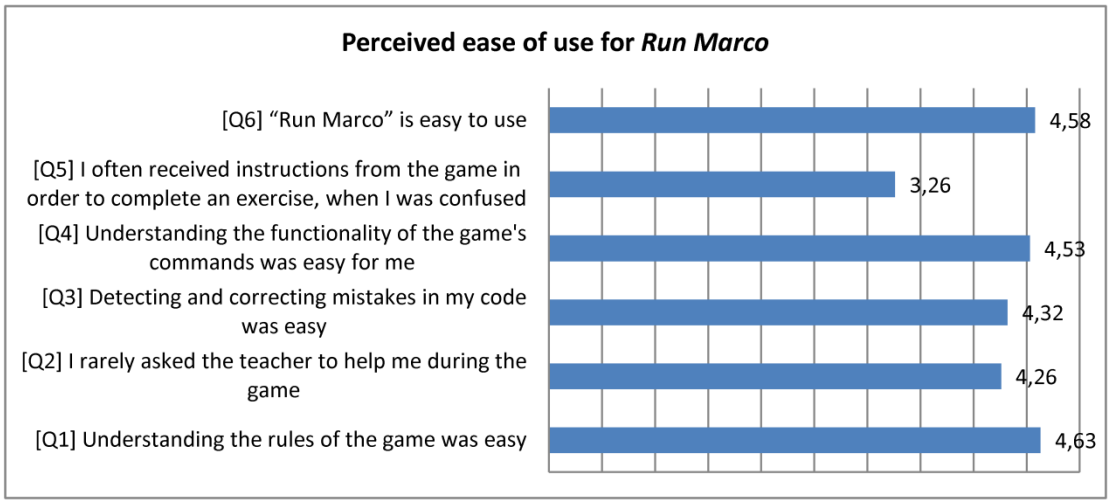

Figure 14. Mean value (scale 1-5) for questions regarding the perceived ease of use for *Run Marco*.

6.3 Students' attitude towards the use of *Run Marco*

Table 6 presents students' answers regarding their attitude towards the use of *Run Marco*. The mean value of students' replies in each question is presented in the chart of Figure 15.

Table 6. Attitude towards the use of *Run Marco*

Question	1 - Strongly Disagree	2 - Disagree	3 - Not Sure	4 – Agree	5 - Strongly Agree	Mean	Standard deviation
Q7. I find that using *Run Marco* to learn programming is a good idea	5.26	21.05	21.05	21.05	31.58	3.53	1.31
Q8. I'd like to have a game for learning programming in the ICT course	5.26	10.53	21.05	36.84	26.32	3.68	1.16

More than half the students (52.63% - Q7: 21.05% agree and 31.58% strongly agree) consider that learning programming using *Run Marco* is a good idea, while 21.05% of them are not sure. Probably this is due to the fact that students did not have any prior experience in game-based learning in other courses and it is difficult for them to

understand how games could be utilized in their context. Moreover, the majority of students (63.16% - Q8: 36.84% agree and 26.32% strongly agree) stated that they would like to use a game in the context of the ICT course, which confirms the dynamics that educational games can have in primary school students.

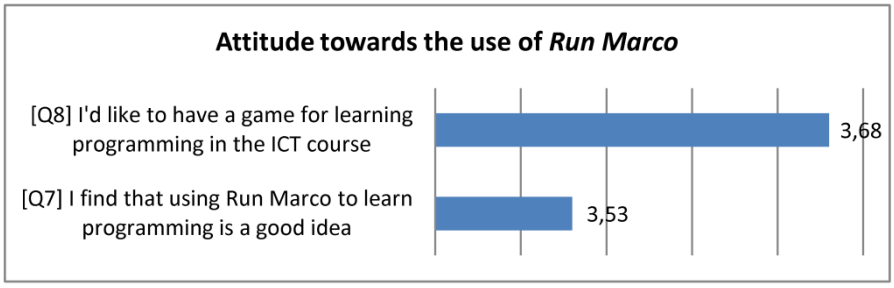

Figure 15. Mean value (scale 1-5) for questions regarding the attitude towards the use of *Run Marco*.

6.4 Perceived usefulness of *Run Marco*

Table 7 presents students' answers regarding their perceived usefulness of *Run Marco*, while the mean value for each question is also presented in Figure 16.

Table 7. Perceived usefulness of *Run Marco*.

	Question	1 - Strongly Disagree	2 - Disagree	3 - Not Sure	4 – Agree	5 - Strongly Agree	Mean	Standard deviation
Q9.	I believe that the lesson is more entertaining with the game "Run Marco"	15.79	15.79	21.05	15.79	31.58	3.32	1.49
Q10.	I think the game gave me the opportunity to participate in the lesson more easily	0	5.26	26.32	21.05	47.37	4.11	0.99
Q11.	I think the game has increased my interest in the ICT course and in computer programming too	21.05	5.26	36.84	31.58	5.26	2.95	1.22
Q12.	I think the game helped me to understand the different ways that a program's commands can be executed	0	5.26	5.26	47.37	42.11	4.26	0.81

Almost half the students (47.37% - Q9: 15.79% agree and 31.58% strongly agree) think that using *Run Marco*, the lesson became more entertaining, while a significant proportion of them (31.58% - Q9: 15.79% strongly disagree and 15.79% disagree) did not seem to have much fun with the game. This may be due to the fact that students of this age (primary school students), who have the experience of modern digital entertainment games with 3D graphics and impressive effects, may have more expectations from an educational game in this field. Also, the majority of students (68.42% - Q10: 21.05% agree and 47.37% strongly agree) declared that the game

gave them the opportunity to participate in the lesson more easily, which shows that the game motivated them. Moreover, students (89.48% - Q12: 47.37% agree and 42.11% strongly agree) believe that the game helped them to understand the programming constructs presented. Finally, the game did not significantly increase students' interest in the ICT lesson but also in programming, as students seem to be divided in this question.

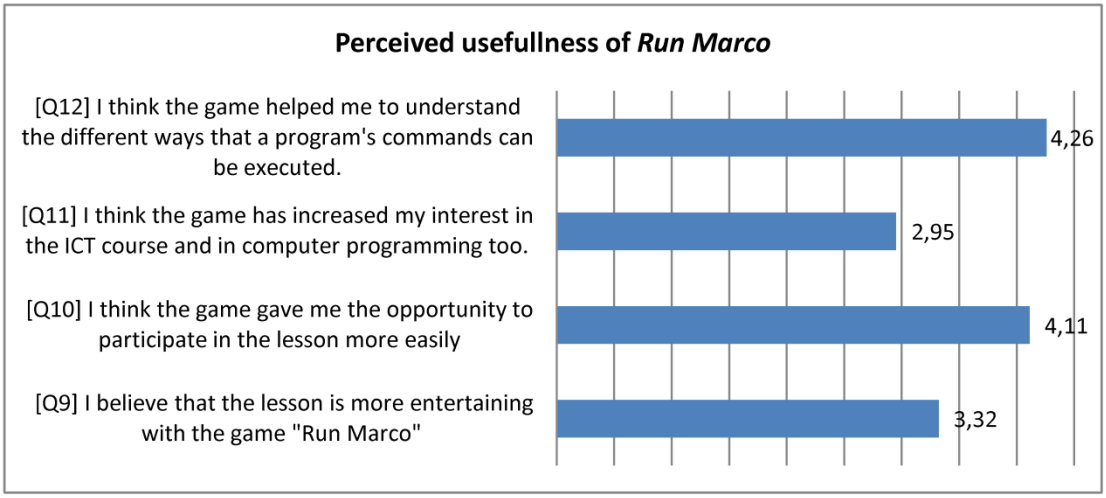

Figure 16. Mean value (scale 1-5) for questions regarding the perceived usefulness of *Run Marco*.

6.5 Behavioural Intention to use *Run Marco*

Table 8 presents students' behavioural intention to use *Run Marco* and Figure 17 provides an overview of the mean value of students' replies.

Table 8. Behavioural Intention to use *Run Marco*.

Question	1 - Strongly Disagree	2 - Disagree	3 - Not Sure	4 – Agree	5 - Strongly Agree	Mean	Standard deviation
Q13. I would like to play again with the game "Run Marco" in the future	21.05	26.32	10.53	21.05	21.05	2.95	1.51
Q14. If there are other similar games to learn programming, I would like to use them.	15.79	10.53	15.79	31.58	26.32	3.42	1.43

Students seem to be divided about the future use of the game and this may be due to the fact that they have more expectations from educational games. However, they are positive about using another similar game to learn programming.

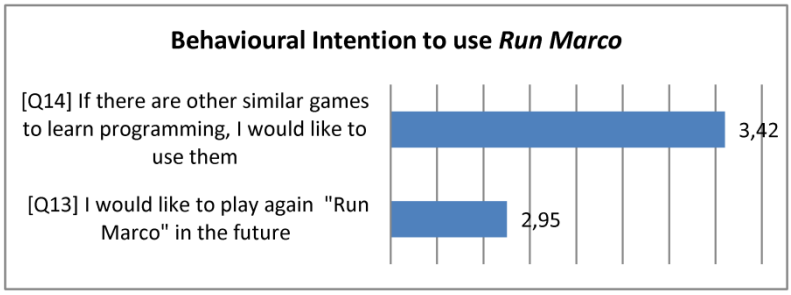

Figure 17. Mean value (scale 1-5) for questions regarding the behavioural intention to use *Run Marco*.

7. Discussion

In this section the results of both the evaluation worksheets and the questionnaire are discussed in the context of each one of the research questions posed in the presented study.

Regarding the *support provided by Run Marco in comprehending programming concepts* (RQ1), the analysis of students' performance on the *evaluation worksheets* and their performance in the game's activities show that teaching basic programming concepts to primary school students can be adequately supported by an educational game environment, like *Run Marco*. This is confirmed also from the results of the two empirical studies mentioned in section 2 where students stated that *Lightbot 2.0* (Mathrani et al., 2016) and *Program Your Robot* (Kazimoglu et al., 2012) helped them in understanding introductory programming concepts effectively.

More specifically in this study students did not encounter difficulties in the activities related to sequence commands, since their performance in both the respective game's activities and the corresponding evaluation worksheet was very good. In loops with a predefined number of iterations, the results are also very satisfactory, as the majority of students (over 60.00%) successfully answered all the questions of the corresponding evaluation worksheet, with two of them even involving nested loops. Finally, students' performance in loops with an undefined number of iterations is quite good, since the majority of students successfully solved three out of the four exercises.

The support provided by the game in comprehending programming concepts was also confirmed by students' replies in the questionnaire. Specifically, students believe that the game helped them understand the basic programming concepts they were taught, enabled them to participate more actively in the lesson and found the game quite entertaining.

Regarding *the acceptance of utilizing educational games for learning programming concepts* (RQ2), the analysis of students' replies in the questionnaire confirms the *acceptance* of educational games by primary school students, as the majority of them found using *Run Marco* in the learning process, as a good idea. Moreover, the

majority of students finds it *easy to use Run Marco* and additionally stated that they would like to use other similar games in the context of the ICT course, which confirms the dynamics that educational games can have in primary school students.

Regarding *the interest raised by Run Marco* (RQ3), the game did not succeed in raising students' interest for the *programming course* to a satisfactory degree, with students' views diverging on this question. This may be due to the fact that students of this age (primary school students), who have the experience of modern digital entertainment games with 3D graphics and impressive effects, may have more expectations from an educational game in this field. Moreover, the activities solved in the context of *Run Marco* (and other similar games) are quite standard and are not characterized by the necessary variety. This fact in combination with the lack of typical game mechanics, such as enemies, the feeling of adventure and/or challenge might have as a side effect that some students lose their interest after playing some levels of the game. The ability of students to build their own levels (scenarios) in the game might raise their interest. It is clear, that all these are assumptions, which however deserve to be researched and also to be taken into account by game designers.

8. Conclusions

In this article a pilot study on the effectiveness of the educational game *Run Marco* for teaching basic programming concepts to primary school students, as well as an investigation of students' acceptance regarding its use in the learning process was presented. In order to investigate this issue a pilot didactical intervention within the "Information and Communication Technologies" (ICT) course taught in primary schools in Greece was designed. The didactical intervention based on *Run Marco* applied three out of the six lesson plans that were designed according to the games' sections. For each lesson plan an *evaluation worksheet* that includes a set of game-related exercises and an auxiliary *activities worksheet*, which provides instructions for solving each level of the game was prepared. The twenty 5^{th} grade students that participated had to play with specific levels of the game and then complete the corresponding *evaluation worksheet.* Based on the teacher's observation the auxiliary *activities worksheet* was particularly useful to students. A *questionnaire* designed in accordance with the Technology Acceptance Model (TAM) was distributed to students after the end of the intervention.

It is clear that the findings of this study were rather positive regarding the utilization of an educational game in the learning process for teaching introductory programming concepts to primary school students. The majority of the participating students found the use of the game *Run Marco* as an effective way to understand the different programming concepts presented in the game, while at the same time this was also confirmed both by their performance in the respective game's activities and their

responses to the exercises of the relevant evaluation worksheets. Students believe that the game added fun elements to the lesson, although it did not fulfill their expectations in this field and moreover the game motivated them to participate in the lesson more actively. Although the game did not increase the interest of the students to a high degree as expected, students stated that they would like to use an educational game in the learning process.

However, the small number of students who participated in this study and the duration (three hours) of this pilot didactic intervention do not allow us to draw safe conclusions about both the effectiveness of the game and students' acceptance for its usage in the learning process. An experimental study including all the six lessons designed for this teaching proposal with a larger number of participants is considered necessary in order to draw more objective and reliable results. The information presented in this article, as well as the instruments used for data collection - including specially designed evaluation worksheets and a questionnaire based on TAM - allow everyone interested in the field to replicate the study. Additionally, further research could include additional studies that examine the effectiveness of other similar games or environments, to teach basic programming concepts to primary school students. Research on the game-play features of educational games for programming that increase the interest of young students is also considered important in order to maximize students' engagement and as a consequence the effect of the game on students' comprehension of programming concepts.

References

Alharbi, S., Drew, S., (2014). Using the Technology Acceptance Model in Understanding Academics' Behavioural Intention to Use Learning Management Systems. In *International Journal of Advanced Computer Science and Applications*, Vol. 5(1), pp 143-155, 13 February 2014. DOI: 10.14569/IJACSA.2014.050120.

Chuttur M.Y. (2009). Overview of the Technology Acceptance Model: Origins, Developments and Future Directions, Indiana University, USA . Sprouts: Working Papers on Information Systems, 9(37). http://sprouts.aisnet.org/9-37

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

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

Evangelopoulou, O., & Xinogalos, S. (2017). MYTH TROUBLES: An Open-Source Educational Game in Scratch for Greek Mythology. *Simulation & Gaming*, Vol. 49, Issue 1, 71-91, DOI: 10.1177/1046878117748175.

Fesakis G., Serafeim K. (2009). Influence of the Familiarization with “Scratch” on Future Teachers’ Opinions and Attitudes about Programming and ICT in Education. *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE-2009)*, Paris, France, 6-8 July, 2009, Vol II, ACM, New York, NY, USA, pp. 258-262

Gibson, B., Bell, T., (2013). Evaluation of games for teaching computer science. *WiPSE November 13 Proceedings of the 8th Workshop in Primary and Secondary Computing Education*, 51-60.

Kandroudi, M., Bratitsis, T. (2016). Teaching programing in small ages with mobile devices through Kodable and ScratchJr: Case Study. *10th Pan-Hellenic and International Conference "ICT in Education", 8th Conference "Teaching of Informatics" of the Hellenic Scientific Association of ICT in Education*, 25-27 September 2016, Ioannina.

Karadeniz, S., Samur, Y., Özden, M.Y., (2014). Playing with Algorithms to Learn Programming: A Case Study on 5 Years Old Children. *9th International Conference on Information Technology and Applications (ICITA2014)*, 1 - 4 July, 2014, Sydney, Australia , http://www.icita.org.

Kazimoglu, C., Kiernan, M., Bacon, L., Mackinnon, L., (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play. *Procedia Computer Science* 9 (2012) 522 – 531.

Laporte, L. and 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 (NordiCHI '16)*. ACM, New York, NY, USA, Article 61 , 10 pages. DOI: https://doi.org/10.1145/2971485.2971499.

Lederer, A. L., Maupin, D. J., Sena, M. P., & Zhuang,Y., (2000). The technology acceptance model and the World Wide Web. *Decision Support Systems*, 29(3), 269-282.

Legris, P., Ingham, J., & Collerette, P., (2003). Why do people use information technology? A critical review of the technology acceptance model. *Information & Management*, 40, 191–204.

Malliarakis, C., Satratzemi, M. and Xinogalos, S. (2014a). CMX: Implementing an MMORPG for Learning Programming. In *Proceedings of 8th European Conference on Games Based Learning*, 9-10 October 2013, Berlin, Germany, 346-355.

Malliarakis, C., Satratzemi, M. and Xinogalos, S. (2014b). Designing educational games for computer programming: A holistic framework. *Electronic Journal of e-Learning*, Volume 12 Issue 3 2014, 281-298.

Malliarakis, C., Satratzemi, M., Xinogalos, S. (2017). CMX: The Effects of an Educational MMORPG on Learning and Teaching Computer Programming, *IEEE Transactions on Learning Technologies*, Vol. 10, Issue 2, 219-235, doi:10.1109/TLT.2016.2556666.

Marco Aedo Lopez, Elizabeth Vidal Duarte, Eveling Castro Gutierrez, and Alfredo Paz Valderrama., (2016). Teaching Abstraction, Function and Reuse in the first class of CS1: A Lightbot Experience. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '16)*. ACM, New York, NY, USA, 256-257. DOI: http://dx.doi.org/10.1145/2899415.2925505.

Mathrani, A., Christian, S., & Ponder-Sutton, A., (2016). PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Educational Technology & Society*, 19 (2), 5–17.

Orehovački, T., & Babić, S. (2015). Evaluating the quality of games designed for learning programming by students with different educational background: An empirical study. In *38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 963-968, IEEE.

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.

Piteira, M., Haddad, S.R. (2011). Innovate in your program computer class: an approach based on a serious game. ITiCSE July 13: *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, Pages 49-54.

Priyanka Surendran (2012). Technology Acceptance Model: A survey of Literature. *International Journal of Business And Social Research* (IJBSR), Vol 2, No: 4, Aug 2012.

Shih-Chi, C., Shing-Han, L. and Chien-Yi, L. (2011) A Literature Review Recent Related Research In Technology Acceptance Model: *Australian Journal of Business and Management Research*, Vol.1 No.9 (2011), 124-127.

Theodoraki, A. and Xinogalos, S. (2014). Studying Students' Attitudes on Using Examples of Game Source Code for Learning Programming. *Informatics in Education*, Vol. 13, No 2, 265-277.

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), 1-7, IEEE.

Xinogalos, S. & Satratzemi, M. (2004). Introducing Novices to Programming: a review of Teaching Approaches and Educational Tools. *Proceedings of the 2ndInternational Conference on Education and Information Systems, Technologies and Applications (EISTA 2004)*, Orlando, Florida, USA, July 21-25, Vol. 2, 60-65.

Xinogalos, S. (2013). Using Flowchart-based Programming Environments for Simplifying Programming and Software Engineering Processes. In *Proceedings of 4th IEEE EDUCON Conference*, Berlin, Germany, 13-15 March 2013, IEEE Press, 1313-1322.

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies*, Volume 21, Issue 3, 559-588, Springer Science+Business Media New York, DOI: 10.1007/s10639-014-9341-9.

Xinogalos, S., Satratzemi, M. & Dagdilelis, V. (2006). An Introduction to object-oriented programming with a didactic microworld: objectKarel, *Computers & Education*, Volume 47, Issue 2, September 2006, 148-171, Elsevier Publishers.

Xinogalos, S., Satratzemi, M., Malliarakis, C. (2017). Microworlds, Games, Animations, Mobile apps, Puzzle editors and more: what is important for an introductory programming environment? *Education and Information Technologies*, Volume 22, Issue 1, 145-176, Springer Sciece+Business Media New York, DOI: 10.1007/s10639-015-9433-1.

Zaharija, G., Mladenović, S., & Boljat, I. (2013). Introducing basic programming concepts to elementary school children. *Procedia-social and behavioral sciences*, *106*, 1576-1584.