Informatics in Education, 2020, Vol. 19, No. 4, 641–668

DOI: 10.15388/infedu.2020.28

BlocklyScript: Design and Pilot Evaluation of an RPG Platform Game for Cultivating Computational Thinking Skills to Young Students

Christodoulos KARAKASIS, Stelios XINOGALOS
Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece
e-mail: mai19024@uom.edu.gr, stelios@uom.edu.gr

Received: January 2020

Abstract. At 21st century Computational Thinking (CT) is considered a fundamental skill that anyone should possess and develop from a young age. Serious games and more specifically educational games (EGs) are a promising means of introducing algorithmic thinking and programming concepts and engaging students through the process of learning. In this article, a new EG called BlocklyScript is presented. BlocklyScript aims to help students develop their CT by learning basic programming concepts, designing algorithms and correcting mistakes. During the designing phase different EGs were taken under consideration and an EG design framework was followed in order to provide a better user experience. The game was evaluated by 10 experienced computer science educators of primary and secondary schools. The positive results of this pilot evaluation show that BlocklyScript is expected to help students understand the basic concepts of CT. However, the game should be evaluated by more teachers and students in order to provide future researchers with safe results.

Keywords: Computational Thinking, educational game, algorithmic thinking, programming concepts.

1. Introduction

During the evolution of Computer Science (CS) many new concepts emerged that were unknown, or at least not widely known, in the past. Computational Thinking (CT) is one of them and at the 21st century it is considered a fundamental skill which is becoming increasingly important. International initiatives, such as the Bebras International Challenge on Informatics and Computational Thinking (Dagienė and Stupurienė, 2016), have contributed in promoting CT in educational institutions all over the world.

Wing (2006) describes CT as a way of solving problems, designing systems, and understanding human behavior that draws on concepts fundamental to CS. She also highlights the importance of developing this skill from a young age since it can help students improve themselves in many different areas such as mathematics, physics and

alchemy. Teachers and especially CS teachers show an intense interest on integrating CT in education (Fessakis and Prantsoudi, 2019) and several approaches have been proposed for achieving this goal. Bell and Bell (2018) propose teaching CT skills, such as decomposition, patterns, abstraction and algorithms, along with music theory and activities. Tonbuloğlu and Tonbuloğlu (2019) propose the use of unplugged coding activities for teaching CT ideas in an appealing and motivating way. Although various approaches have been proposed, CT skills are usually approached through the teaching and learning of programming and all the activities that it encompasses (Combéfis *et al*., 2016; Giannakoulas & Xinogalos, 2020). However, learning programming from a young age can be challenging, since students are not familiar with programming concepts and they often lose their incentives (Zaharija *et al*., 2013).

Educational games (EGs) are a promising means of introducing algorithmic thinking and programming concepts and engaging students in the learning process. The main reason why EGs are considered a beneficial educational method derives from the motivation that EGs can offer to young learners (Stege *et al*., 2011). It is generally accepted that the key of active learning is the students' motivation. Furthermore young students are more confident with the concept of "playing" rather than reading a traditional book. Svinicki (1999) showed that traditional teaching methods do not motivate students to participate in the classroom. For these reasons many EGs were developed in order to provide incentives to students and engage them in learning programming and acquiring CT skills (Giannakoulas & Xinogalos, 2018; Giannakoulas & Xinogalos, 2020; Kazimoglu *et al*., 2012).

Moreover, EGs for introducing programming are capable of promoting self-studying through theory instructions, playful hands-on programming activities, immediate feedback and scaffolding levels. Self-learning in the limelight couldn't be of greater importance since the lockdown of billion students at their homes due to COVID-19 pandemic. UNESCO (2020) published that over 140 countries around the world were led to close (country-wide) their schools affecting the education progress of students in order to confine the spread of the virus. To help those countries UNESCO (2020) provided a list of distance learning solutions which includes among others digital learning and self-directed learning content such as EGs.

In this article, a new EG called BlocklyScript is presented, which aims to help students develop their CT skills by learning basic programming concepts, locating patterns, designing algorithms and correcting mistakes through clear and specific rules. BlocklyScript is a 2D online platform game which incorporates a custom made visual based programming language and presents programming concepts, such as sequence commands, loops, if-statements, methods and variables. Unlike most of the EGs that already exist on the Internet and are characterized as puzzle games, BlocklyScript attempts to follow a different approach. It combines elements of an RPG (Role-Playing Game) by controlling an explorer (male or female) and learning new abilities (laser beam) in a typical platform game. It also incorporates enemies (flying aliens) and obstacles (doors, spikes, walls) which are expected to maintain undiminished the interest of the player. Furthermore it provides a rich rewarding system (score, badges, time) which captures the user's attention. Lastly it helps students to recover from their mistakes with the assistance of articulate tips and advices in the case of an error at the end of each attempt.

Debriefing is a very important feature of any serious game and consequently any EG. "*Debriefing is the processing of the game experience to turn it into learning (to paraphrase Dave Kolb)*" (Crookall, 2010, p. 907). Crookall (2010) emphasizes on the importance of debriefing for learning and for establishing simulation/gaming as a discipline. At each level in BlocklyScript the user has to design an algorithm based on the theory s/he reads and immediately after its execution the player is informed about the correctness of his/her solution. This is where debriefing takes place and in the case of a wrong solution extra tips are provided to the player in order to help him/her understand the mistakes. Furthermore the player can access anytime the theory learned through previous levels and iterate through them. In case of active learning inside a classroom debriefing takes the form of teacher's interaction with the students. The teacher can check the results table at the end of the game for each student and learn which theory concept was the hardest for the students to comprehend. In both cases BlocklyScript can support either students in self-assessment or teachers on improving the way of presenting programming concepts.

The overall methodology of the study presented in this article consisted of three phases. Firstly, a *literature review* on studies investigating the effects of game-based learning (GBL) and EGs on teaching and learning programming and cultivating CT skills targeted to young students was carried out. This was followed by *play-testing* and *comparatively analyzing four EGs* using factors that were recorded in the literature. In the second phase, BlocklyScript was designed utilizing the results of the literature review and the comparative analysis carried out, as well as the *Educational Games Design* model proposed by Ibrahim and Jaafar (2009). In the third phase, a *pilot evaluation* of BlocklyScript took place utilizing a *survey*. Specifically, the game was play-tested by volunteer teachers who filled in an online questionnaire that was based on the MEEGA+ model (Petri *et al*., 2018). The main research question of this pilot evaluation was:

> *What are teachers' perceptions on the usability and user experience of BlocklyScript?*

The investigation of this research question was deemed necessary in order to: validate the design of BlocklyScript; and draw conclusions on the acceptance of BlocklyScript by teachers, as well as their willingness to utilize it in the classroom as a tool for introducing young students to programming and cultivating CT skills. Based on the results, we could either make the necessary changes to the game or make it available to interested teachers for utilizing it in the classroom and studying students' actual conceptual learning within the game.

The rest of this article is organized as follows. In section 2, a literature review of relevant work and a comparative analysis of existing EGs are presented. In section 3 the design and analysis of the game are presented. In section 4 the methodology of the pilot evaluation of BlocklyScript is presented, followed by the results, the limitations of the study and a critical discussion of the results. Next, some conclusions are drawn and implications for researchers and practitioners are presented. In the last section suggestions for future research are proposed.

2. Literature Review

2.1. *Game-based Learning Pedagogy*

Marc Prensky back in 2003 highlighted how the learners have changed and how digital games can be used for satisfying their learning needs (Prensky, 2003). Since then research on the possibilities of the so called serious games (Zyda, 2005), and especially their subcategory of educational games, for an enhanced teaching and learning of various subjects has attracted the interest of researchers and instructors.

In a more recent study, Rebah (2019) highlights three pedagogical virtues of serious games. The first virtue, as mentioned earlier, is the ability to capture the intrinsic motivation of learners. A student may forget that s/he is trying to acquire new knowledge and stay motivated by focusing on how to complete the game challenges. The second major benefit is described as "situated learning". Learners must find the best solution to solve a situation-problem by focusing on a single task at hand and therefore improving their productivity. The third virtue refers to the opportunity to learn from mistakes. A well designed serious game can offer immediate feedback to the learners in case of an error, helping them to find the right solution to the existing problem.

Various research studies have been carried out on EGs whose educational objective is the introduction of programming concepts to novices and in particular to young students.

"Aladdin and his flying carpet" is a 3D EG that teaches solution efficiency, pattern recognition, debugging and decomposition to its pupils. The game was evaluated by 107 sixth grade pupils with no prior knowledge of programming (Jakoš & Verber, 2017). The results showed that after playing the game (posttests) 97% of the pupils met the minimum knowledge standards, as they are defined on the curriculum of the Computer Science Course for primary schools in Slovenia. In the pretest phase, the percentage of the students that met the minimum knowledge standards was 25%.

"Minerva" is a 2D puzzle game designed for teaching basic programming concepts to elementary school students without coding. A mixed-method evaluation was conducted with 32 Korean 6th grade students playing the game and 32 6th grade students who studied the same concepts using handouts (Lindberg & Laine, 2018). Initially the results regarding the learning experience of the game-based learning (GBL) group showed that the students increased their knowledge about computer programming by completing challenging tasks and gained a sense of achievement through the game. In addition, the GBL group trowed that "Minerva" was an exciting learning experience (75%) and useful (91%) for their career. In terms of retention, both groups performed well. On one hand, the GBL group performed better at answering specified tasks (issuing commands, loops, conditions), but on the other hand the second group was more efficient at more complex tasks (combination of theory).

"MiniColon" is an EG that aims at teaching young children the preliminary concepts of programming (sequential blocks, conditional blocks and iterative blocks) by controlling an avatar with a Kinetic Sensor device (Ayman *et al*., 2018). Similar to

"Minerva" the game was evaluated by 8 young students who played the game and 7 students who approached the basic programming concepts through a traditional education method. Both groups answered a questionnaire before and after learning the new concepts and the learning gain was calculated by subtracting the number of correct answers in the pretest from the posttest quizzes. The results indicated that the learning gain and the immersion achieved by the GBL group were significantly higher compared to the second group.

Giannakoulas and Xinogalos (2018) presented a pilot study on the effectiveness and acceptance of the EG "Run Marco!" for teaching programming concepts to 19 primary school students in Greece. The students were introduced to sequence commands, loops with a predefined and an undefined number of iterations by completing the corresponding levels of the game and using supplementary educational material prepared by the researchers. The concepts were divided into 3 lessons and at the end of each lesson, students filled an evaluation worksheet. In addition, after completing all the lessons, a questionnaire based on the widely known Technology Acceptance Model (TAM) was answered by the students regarding the perceived ease of use and usefulness of "Run Marco!" as well as the students' attitude towards the use of it. The results of the pilot study showed that "Run Marco!" supported the students in comprehending the aforementioned concepts, while it was accepted by the majority of them as a useful alternative learning process. However the game was unable to raise students' interest at a high degree and based on the authors this might be due to the lack of typical game mechanics that are present in purely entertainment games such as enemies and the feeling of adventure or challenge.

Danks *et al*. (2019) present how "Code Combat" has been used for at least one month by 170 teachers across the United States of America (mainly in California) for introducing 1st–12th grade students to basic programming concepts. "Code Combat" was used by 46% of teachers as a supplementary resource to the existing computer science curricula, while 36% used the EG as the core curriculum resource. Findings reveal that 90% of teachers agreed or strongly agreed that "Code Combat" keeps students engaged and 97% agreed or strongly agreed that the benefits students gain from the EG are worth their time. More specifically the EG allows teachers to support students' problem-solving abilities and help them when they are stuck, it helps them teach students how to become strong coders by providing effective coding practices and it nurtures students' curiosity in the classroom.

Du *et al*. (2016; 2018) conducted a study in order to investigate the effect of the Hour of Code tutorials such as Minecraft and Angry Birds on students' attitude toward programming. The study was conducted at one Midwestern university for 4 years (2014–2017) in an introductory computing course and a data set (quantitative and qualitative) of 255 college students, novices to programming, was selected over the years. The quantitative data analysis (Du *et al*, 2016) of 116 participants showed that the positive change of students' attitudes toward programming was statistically significant and the awareness of the importance of programming was increased. In addition 97 open-ended qualitative comments (55 from the pre-survey and 42 from the post-survey) were collected till the year of 2017 indicating the same results. Most students (64%)

had a negative attitude towards programming before playing the EG which changed after playing it with nearly 70% of positive comments. Although this study refers to higher education students, it was included in our review since the specific game can be used by young students as well.

Most of the reviewed studies focus on the final learning outcomes and the advantages of the EGs they utilized, while pedagogical issues such as practical implications for integrating the use of EGs in the classroom are less frequently addressed (Giannakoulas and Xinogalos, 2018). Also, some studies refer solely on the acquisition of programming skills and do not make a clear reference on CT skills, such as algorithmic thinking, abstraction, decomposition, modularity and generalization that are considered as fundamental CT skills (Atmatzidou and Demetriadis, 2016). However, it is clear that programming activities included in EGs have the potential to cultivate CT skills in a straightforward manner. For example, in the context of an EG students typically have to implement algorithms for guiding the protagonist of the game in carrying out various tasks, thus cultivating *algorithmic thinking skills*. As the game progresses, the tasks become more complicated and require breaking down the problem into sub-problems that are easier to deal with, thus leading to the cultivation of *decomposition* and *modularity* skills. Moreover, some tasks or sub-problems require the *recognition of a pattern* that is interpreted to a piece of code which is afterwards repeated using a loop. In some occasions, a task or sub-problem (either representing a pattern or not) has to be used in different parts of the whole program or even more has to be parameterized (for example, moving the player for a different distance), thus needing to define functions which cultivate *abstraction* and *generalization* skills.

2.2. *Educational Games for Programming*

Contemporary programming languages fall under two major categories. The first category includes text based programming languages, while the second one includes visual based programming languages. The influence that each one of these two categories of programming languages can have on young students is different. In a study comparing the influence of the aforementioned categories the results showed that visual based programming is a better approach for teaching the basic concepts of programming to novices (Saito *et al*., 2017).

For the design and development of BlocklyScript it was imperative to study existing online EGs aiming at developing CT skills to young students. The games studied were not limited only to those that support visual programming, but also to text-based ones. The games selected were “Code Combat” which supports a text-based editor, “Hour of Code: Minecraft” and “Run Marco!” which support a visual based language and “Rapid Router” which includes both ways of programming. The specific EGs were selected for the comparative analysis based on the following criteria: they are freely available and can be play-tested by anyone in order to gain experience in both their mechanics and their underlying pedagogy; they are widely known and referenced; they constitute representative examples of programming EGs that are based on a text-based programming

language, a visual one or a combination of both. Interested readers can find an extended review of twenty-two programming EGs targeted to young students in (Giannakoulas & Xinogalos, 2020).

Code Combat is a 2D web-based game which aims to teach programming to young students in Python, JavaScript or CoffeeScript. It is categorized as an RPG (Role-Playing Game) and the theory is divided into many different levels. Each level is composed of the game area and the text-based editor. The game area includes a panel where the game character exists and the text-based editor is the area where the player interacts with the game character. Each level includes specific and clear goals and a hint button to help the user recover from a mistake. Furthermore Code Combat allows the user to choose between many characters with different statistics, it includes a variety of items that a character can carry and rewards the player with achievements and gems with witch the player can buy a better equipment (available at: `https://codecombat.com/premium`).

Hour of Code: Minecraft is a 2.5D web-based game, which includes four different game scenarios. Each game scenario consists of 10–14 levels and aims to teach a few programming concepts in an hour. The structure of the levels is the same as Code Combat with the difference that the editor is visual-based and includes interlocking blocks based on the Blockly library. By reading the theory and watching some introductory videos the user can comprehend the utility of each block and then s/he can drag and drop the blocks and connect them with each other in order to design an algorithm. The designed algorithm should, afterwards, guide the game character (male or female) in order to achieve the objectives of the corresponding level (available at: `https://code.org/minecraft`).

Run Marco! is a puzzle game that aims to teach basic programming concepts to novices with the help of a visual based language. The user can choose between a male and a female character and guide him/her through a jungle in order to find his/her friends. The user commands the character how to move by dragging and dropping blocks and tries to find the correct path which will lead to the final destination. The game rewards the player at the end of each attempt in every level (available at: `https://www.allcancode.com/hourofcode`).

Rapid Router is a 2D web-based puzzle game which aims to introduce the basic concepts of programming through 109 different levels. The user controls a van through a visual based language based on Blockly and attempts to deliver food supplies to the houses. The twist that Rapid Router incorporates compared to Hour of Code: Minecraft and Run Marco! is that at the last 29 levels it introduces novices to the programming language of Python by translating the blocks into text-based commands (available at: `https://www.codeforlife.education/rapidrouter/`).

The four EGs presented have the same aim, to teach programming concepts to novices. However there are some differences in the educational material that each one of them incorporates. In Table 1 we present the programming concepts covered by these EGs. Code Combat and Rapid Router include all of the programming concepts as shown is Table 1, while the concepts of object oriented programming and variables are not supported by Run Marco! and Hour of Code: Minecraft. Furthermore Hour of Code: Minecraft and Run Marco! do not incorporate exercises with nested ifs and functions respectively.

Table 1
Programming Concepts

Concept	Code Combat	Hour of Code: Minecraft	Rapid Router	Run Marco!
Sequence commands	✓	✓	✓	✓
Simple if	✓	✓	✓	✓
If-else	✓	✓	✓	✓
Nested if	✓	×	✓	✓
For loops	✓	✓	✓	✓
Do while or Repeat until	✓	✓	✓	✓
Functions	✓	✓	✓	×
Object oriented programming	✓	×	✓	×
Variables	✓	×	✓	×

In addition, some basic features that an EG must meet have been studied. These features can be divided into five categories:

- Scenario of the game.
- Game mechanics.
- Program execution, test and debugging tools.
- Supported platforms, and more specifically the game's availability on iOS, Android and online platforms.
- CT dimensions or else CT skills that are cultivated through the EG. The estimation for supporting or not the cultivation of each examined CT skill by each one of the EGs is based on the analysis provided at the end of section 2.1.

The study of the four EGs also included players' prerequisite knowledge requirements for each game, the programming language(s) supported and enhanced features such as support for online classrooms and creation of new game levels. The features used in the comparative analysis where based on the works by (Giannakoulas & Xinogalos, 2018; 2020), which in their turn utilized metrics that are based on specialized design frameworks for EGs. Table 2 summarizes the aforementioned information for the EGs that were studied.

3. Analysis and Design of BlocklyScript

In this section the analysis and design of BlocklyScript are presented. The analysis of BlocklyScript is related to some basic data of the game such as the instructional objectives, the targeted audience, the preparation and set-up time, the playing time, and the game objectives. After the analysis of the game, the selected design framework is introduced which allowed the developer to incorporate the basic features that characterize a web-based EG.

Table 2
Game characteristics

	Feature	Code Combat	Hour of Code: Minecraft	Rapid Router	Run Marco!
	Prerequisite knowledge	No	No	No	No
	Player's educational degree	Primary school	Primary school	Primary school	Primary school
Scenario	**Variety of activities**	Designing an algorithm for moving/ guiding the avatar	Designing an algorithm for moving/ guiding the avatar	Designing an algorithm for moving/ guiding the avatar	Designing an algorithm for moving/ guiding the avatar
Scenario	**Select character option**	✓	✓	×	✓
Scenario	**Enemies**	✓	✓	×	×
Scenario	**Activities offered**	Design an algorithm and assigning values to variables	Design an algorithm	Design an algorithm	Design an algorithm
Game mechanics	**Scaffolding/ Support**	Instructions per level in text format	Instructions per level in text format & introductory videos	Instructions per level in text format	Instructions per level in text format
Game mechanics	**Score**	✓	×	✓	×
Game mechanics	**Reward the victory of every level**	✓	✓	✓	✓
Game mechanics	**Reward best solution**	×	✓	✓	✓
Game mechanics	**Achievements**	✓	×	×	×
Program execution Test and debug	**Programming language**	Python, JavaScript, CoffeeScript	Visual programming with Blockly	Visual programming with Blockly	Visual programming with Blockly
Program execution Test and debug	**Highlighting the command being executed**	✓	×	✓	✓
Program execution Test and debug	**Execute program at different speeds**	✓	×	✓	×
Program execution Test and debug	**Feedback in case of an error**	✓	×	✓	×
Program execution Test and debug	**Testing and debugging**	Executing the program	Executing the program	Executing the program or step by step execution	Executing the program
Program execution Test and debug	**Pause program execution**	✓	×	✓	×
	Online classrooms	✓	✓	✓	×
	Create new levels	✓	✓	✓	×
Available platforms	**Web**	✓	✓	✓	✓
Available platforms	**Android**	×	×	×	✓
Available platforms	**iOs**	×	×	×	✓
CT Dimensions	**Algorithmic thinking**	✓	✓	✓	✓
CT Dimensions	**Decomposition & modularity**	✓	✓	✓	✓
CT Dimensions	**Patterns recognition**	✓	✓	✓	✓
CT Dimensions	**Abstraction & generalization**	✓	✓	✓	×

3.1. *Analysis of BlocklyScript*

Instructional Objectives of BlocklyScript

The instructional objectives of BlocklyScript can be summarized as follows:

- Promotes the active learning and participation inside the classroom.
- Promotes self-learning and self-assessment outside of the classroom.
- Presents, explains and familiarizes students with the basic concepts of programming through plain, simple and descriptive text, images and examples, as well as with the support of the visual programming library of Blockly. The theory includes sequence commands, simple if, if-else, nested if, for, repeat while, repeat until, methods and variables.
- Improves students' CT skills by designing interactive algorithms in order to control the protagonist of the game.

Target Audience

The target group of this game is primary school students who try to learn the basic concepts of programming for the first time. In addition it encourages anyone who wants to improve his/her CT skills by designing algorithms.

Preparation and Set-Up Time

This is an online 2D platform game that runs in a browser. So the only requirements are a computer with access to the Internet and an installed browser. The game was tested in two popular browsers, Mozilla and Chrome.

Playing Time

The playing time depends mainly on the player's abilities in comprehending the presented programming concepts and the process of designing an algorithm for solving a problem. The estimated time for a student to win all the levels is between 2–4 hours. For this reason the game saves user's progress inside a database so s/he can play it anytime and continue from the last saved point.

Source Code and Installation Tips

The source code of BlocklyScript along with installation tips is available at the following URL: `https://github.com/CKar95/BlocklyScript`

Game Objectives – Game Scenario

BlocklyScript is a 2D platformer action game in which the player controls the movement of a Blocklyland adventurer. The player is challenged to guide the adventurer by designing algorithms through the levels of Blocklyland in order to collect all the stars that fell suddenly from the sky and threatens the planet with eternal darkness. The adventurer will also have to collect some balloons during his/her journey with which s/he will send the stars back to the sky at the end of the game. Before the player starts

playing the game s/he will have to create an account which will identify him/her and save his/her progress inside the game.

Game Flow

- After the user's successful login, the game is available. The game is divided into 12 levels and each one of them contains a new programming concept which is integrated into visual blocks. The visual blocks comprise the available commands with which the player interacts and commands the adventurer to take action. The player will have to unlock the initial levels first and then move forward to unlock the rest of them as shown in Fig. 1.
- The design of each level contains four different areas (Fig. 2). The top left corner represents the game area. The top right corner represents the Blockly area which contains the available block commands. The bottom left corner includes the instructions/theory of the corresponding level. The bottom right corner is the display area of the program execution results.
- Firstly the player will have to read the instructions area which contains the goals of the level and useful information about new obstacles and the functionality of the block commands (Fig. 2).
- After comprehending the instructions the player will have to analyze the task, locate potential patterns, and design an algorithm based on the block commands s/he learned at the current and previous levels (Fig. 2). The user can access the theory of the running and previous levels via a help button.
- At the end of the execution of each attempt the user is immediately informed about the correctness of his/her answer and rewarded (points, badges) accordingly (Fig. 3). In the case of a victory the quality of the solution is checked based on

Fig. 1. The first two completed levels of BlocklyScript.

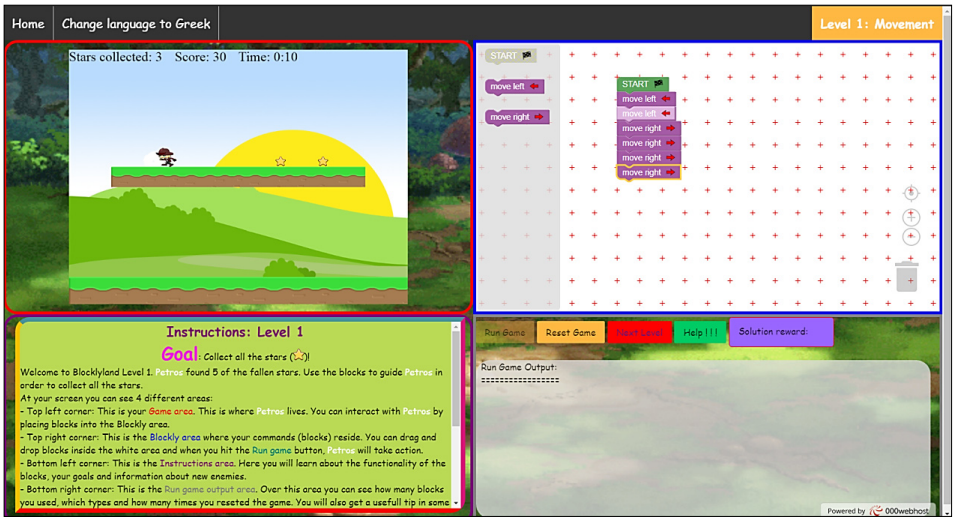

Fig. 2. First level of BlocklyScript.

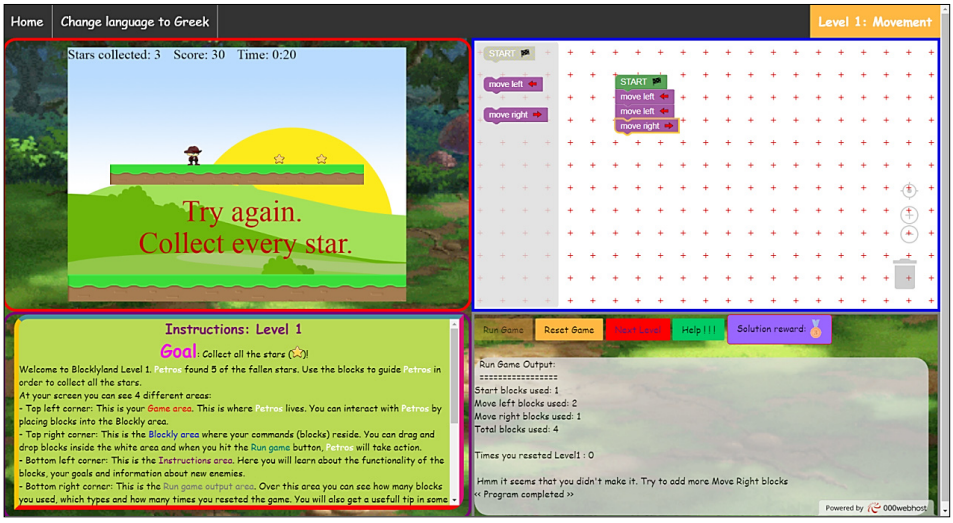

Fig. 3. Level 1 failed attempt tips and reward.

the number of blocks used. In the case of a defeat extra tips, in exchange for score points, are presented which can help the player comprehend easier the new theory and correct his/her algorithm in order to find the best solution.

- As the game progresses new obstacles and enemies appear that block the adventurer's path. On the one hand obstacles can take the form of spikes, walls, doors or the space between two platforms. On the other hand moving enemies take the form of a flying yellow alien who is attracted to the light of the fallen stars and attempts to protect them from the adventurers. The levels of BlocklyScript are presented in Fig. 4.

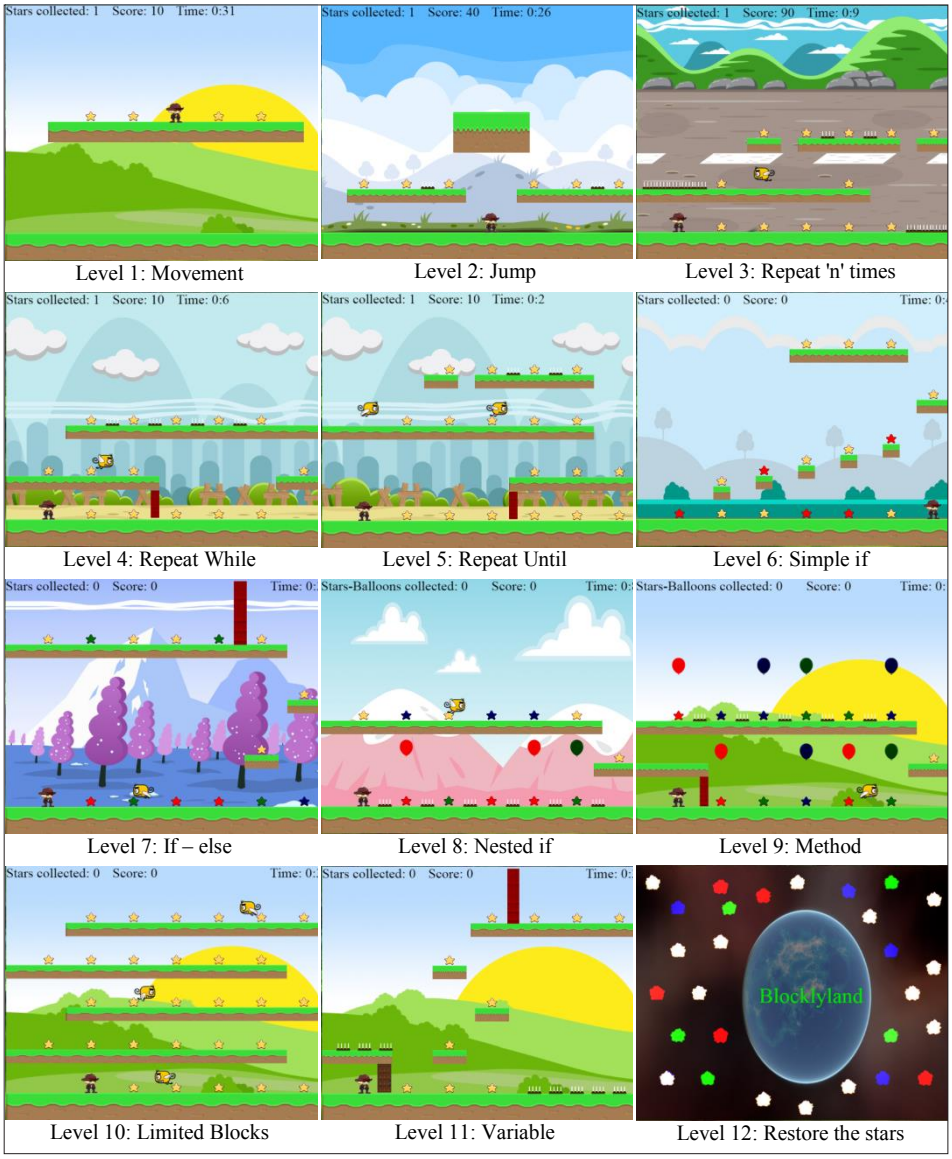

Fig. 4. The levels of BlocklyScript.

- After completing all the levels the player can restore the stars back to the sky and fulfill his initial goal presented by the scenario. Furthermore the player can check useful statistics relating to his/her answers for each level.

Finally the game allows the player to iterate through the levels s/he already won without any restriction regarding the number of his/her attempts and choose between English and Greek.

3.2. *Design of BlocklyScript*

The combination of a well-designed EG with its pedagogical objectives optimizes its benefits and effectiveness in education. For this reason, an extensive research was carried out to select the appropriate design framework that meets the needs of BlocklyScript. Malliarakis, Satratzemi & Xinogalos (2014a) presented several of the available EGs design frameworks, indicating their features. On this basis, the design model proposed by Ibrahim & Jaafar (2009) was selected. This model was designed to support student's self-instruction and self-assessment within the game. The *Educational Games Design* model is based on three main axes. The axes are:

- Game Design.
- Pedagogy.
- Learning Content Modeling.

Game Design (Table 3) relates to the usability, the multimodality and the fun that the game offers. The usability of the game is based on its satisfaction, efficiency and effectiveness and should be tested through the ISO 9241 usability prototype (Pinelle and Wong, 2008). Multimodality is very important for EGs because it combines multimedia, such as text, graphics, sound, video and animation with which students interact and receive immediate feedback (Ibrahim & Jaafar, 2009). Lastly the entertainment that the EG offers, provides motivation and commitment to players. In Table 3 the design decisions that were made for BlocklyScript in terms of the factors included in the Game Design axis of the model are presented.

Table 3
Game Design

Factor (Ibrahim & Jaafar, 2009)	Design decision
Usability	
- *Satisfaction*	✓ The game environment is user-friendly and easy to navigate. ✓ The interesting scenario combined with attractive graphics and sounds provide aesthetic satisfaction to the user. In addition, saving the score, time and badge upon completing a level gives the player joy in seeing his/her effort rewarded. It also allows the users to play the game whenever they have free time by storing their progress in a database. ✓ The encouragement to identify the optimal solution creates a sense of satisfaction for the player.
- *Efficiency*	✓ The game is provided for free on the web and is supported by the most popular browsers. ✓ The theory is presented through a simple and concise text combined with pictures and simple examples that apply the theory. Also, the categorization of block commands and color matching makes it easier for the student to understand their functionality.

Continued on next page

Table 3 – continued from previous page

Factor (Ibrahim & Jaafar, 2009)	Design decision
- *Effectiveness*	✓ The help button gives access to previous levels instructions/theory and reminds students of all the concepts they learned. ✓ Introducing a single concept per level contributes to its more effective learning. ✓ Checking the player's solution and providing feedback on its correctness and quality helps the player understand exactly where s/he may have gone wrong.
Multimodal	
- *Multimedia and interaction*	✓ Incorporating objects for collection (stars, balloons), avoiding obstacles (spikes, walls, doors), dealing with enemies (flying aliens) and exploring secret passwords keep student's interest undiminished. ✓ Relaxed and joyful background music helps the user to think and relax in his/her attempt to design the appropriate algorithm to solve the level. ✓ The appearance of appropriate messages and sound at the end of the user's algorithm execution separates the successful from the wrong solution. ✓ The user interacts with the explorer by dragging and dropping the available command blocks in the Blockly editor and connecting them appropriately. In addition there are various types of blocks that require user input. This input can be the selection of the appropriate color or condition, the block's formatting and the typing of an alphanumeric.
Fun – Challenge	
- *Clear goals*	✓ The objectives at each level are clear and appear in the level's instruction area. The main objective is for the explorer to collect all the available stars and/or balloons in the level with the help of the player's algorithm.
- *Uncertain outcome*	✓ The player has to find the solution of the level in order to continue to the next levels and to see the results of his/her solution.
- *Self esteem*	✓ The presence of scoring and badges, the appearance of tailor-made messages to find the optimal solution and restarting the levels without any restrictions boost the player's confidence.

Pedagogy concerns the final degree of achievement of the learning results. The game will be designed based on the first three levels of Bloom's educational outcomes classification (knowledge, understanding, application). In addition, theory should motivate students and support autonomous learning, while the EG should develop the student's problem-solving abilities. The design decisions made for satisfying the pedagogical factors of the Educational Games Design model in BlocklyScript are presented in Table 4.

Learning Content Modeling argues that game lessons should be designed in a way that supports self-learning and achieves its teaching goals. In addition, the presentation of the educational material should be done in a smooth and progressive manner in order to stimulate the interest of the player. In Table 5 the way that these requirements were approached during the design of BlocklyScript is presented.

Table 4
Pedagogy

Factor (Ibrahim & Jaafar, 2009)	Design decision
- *Learning outcomes*	✓ The user will learn basic programming concepts such as sequential execution, loops, ifs, restricted use of commands and variables and acquire CT skills, such as analyzing problems, locating patterns, designing algorithms.
- *Motivation Theory*	✓ As players progress, they enrich their knowledge of the basic concepts of programming. The theory is presented through a comprehensive text along with illustrations and examples of the application of the block commands. ✓ The user is interested in the theory s/he has already learned because s/he uses it in conjunction with the concepts of the new level to find the solution.
- *Self-learning*	✓ The game does not require from the user any pre-existing knowledge on programming. At each level the user tries to assimilate a new programming concept. ✓ The game allows the user to look at the theory of each level as well as to restart it without limiting the algorithm's redesign efforts.
- *Problem solving*	✓ At each level the users apply the theory they learned to design an algorithm and solve the level by guiding the explorer. ✓ With the help of the «Program output» area the user can understand exactly where s/he made a mistake and read helpful tips on how to apply the theory correctly and even devise a better quality solution/program.

Table 5
Learning Content Modeling

Factor (Ibrahim & Jaafar, 2009)	Design decision
- Syllabus matching	✓ Similar EGs that aim to teach programming concepts to young students were studied and the educational content of BlocklyScript was designed based on them.
- Scaffolding	✓ The flow of theory presentation is smooth. Initially the students learn sequential execution. Then they learn the repeating loops. Afterwards the student comprehends the "if" commands and at the last levels s/he learns about the methods and the variables. The user can even spend as much time as s/he wants on each level to read the theory and gradually build on the existing knowledge.

4. Pilot Evaluation Methodology

4.1. *Research Method*

The pilot evaluation of BlocklyScript is part of a broader research on the potential effectiveness of game-based learning and EGs on teaching and learning programming and cultivating CT skills to young students. The *motivation for the research* presented

in this article was to review relevant literature in order to investigate whether there is a need for another EG for programming and CT. BlocklyScript, in contrast with most of the EGs of this kind that are puzzle games, is an RPG platform game that tries to encompass more entertaining game mechanics, such as enemies and obstacles, in order to motivate students. Furthermore it provides a rich rewarding system (score, badges, time) and a mechanism for reporting articulate tips and advices in the case of an error in order to assist students in recovering from errors and comprehending the semantics of the underlying concepts.

The main *research question of the pilot evaluation* of BlocklyScript was:

> *What are teachers' perceptions on the usability and user experience of BlocklyScript?*

The *purpose of the research study* presented in the next paragraphs was twofold: firstly, to validate the design of BlocklyScript in terms of teachers' perceptions regarding its usability and the experience of the students that will use the game; and secondly, to draw conclusions on the acceptance of BlocklyScript by teachers, as well as their willingness to utilize it in the classroom as a tool for introducing young students to programming and cultivating CT skills. This would give us the chance to make BlocklyScript available to the educational community for usage in the classroom or making the necessary corrections prior its usage and evaluation by students.

In order to investigate the aforementioned research question a survey research approach was adopted and the results were quantitatively analyzed. Specifically, an attempt to attract volunteer teachers for play-testing the game and evaluating it was made. For this purpose, an invitation was sent by email to Information Technology educators of primary and secondary schools. Moreover, this invitation was posted on a relevant forum by an Informatics public school counselor at the district of East Thessaloniki and Chalkidiki in Central Macedonia. This forum is intended for teachers with an interest on Information and Communication Technologies (ICT) in education and the pedagogy of teaching and learning programming to and by students respectively. Participants were given a fifteen-day interval to play the EG online by following the link in the invitation and then evaluating it by responding anonymously to a special designed questionnaire.

4.2. *Participants*

A total of ten Information Technology (IT) teachers of primary and secondary schools responded to the invitation to play test and evaluate BlocklyScript. The participants played BlocklyScript at the leisure of their home and then evaluated its expected contribution to students' education. The recipients of the invitation, and consequently our participants, were all expected to have knowledge on teaching programming since it is a major subject both at primary and secondary education. However we could not know in advance whether they would have any experience on utilizing EGs in education. We must note that we expected to attract more participants. The small number of participants

might be attributed to the fact that the educational material and the GUI of the initial version of BlocklyScript was offered only in English. This might have deterred teachers, since they could not easily use a game that does not support the mother tongue of their students. Moreover, it might be true that teachers are still hesitant in utilizing EGs in their classrooms.

The demographics of the participants are presented in Fig. 5. Out of the ten IT educators that participated in the pilot evaluation, 30% teach to elementary school students and the rest 70% to high school students. It is worth noting that 70% are over 40 years old (40–50: 60%, 50 or older: 10%), indicating that they have several years of experience in student education. Also, it must be noted that all the teachers have utilized educational games in the classroom and half of them have used more than five educational games (from 5 to 10: 20%, over 10: 30%) and consequently can be considered qualified to evaluate the pros and cons of such a game. Moreover, 50% of the teachers have developed their own educational game or customized an existing one and know how to design it in order to achieve its goals. So despite the small sample of the study participants, they all have experience in utilizing educational games in the classroom and half of them in developing and/or customizing educational games.

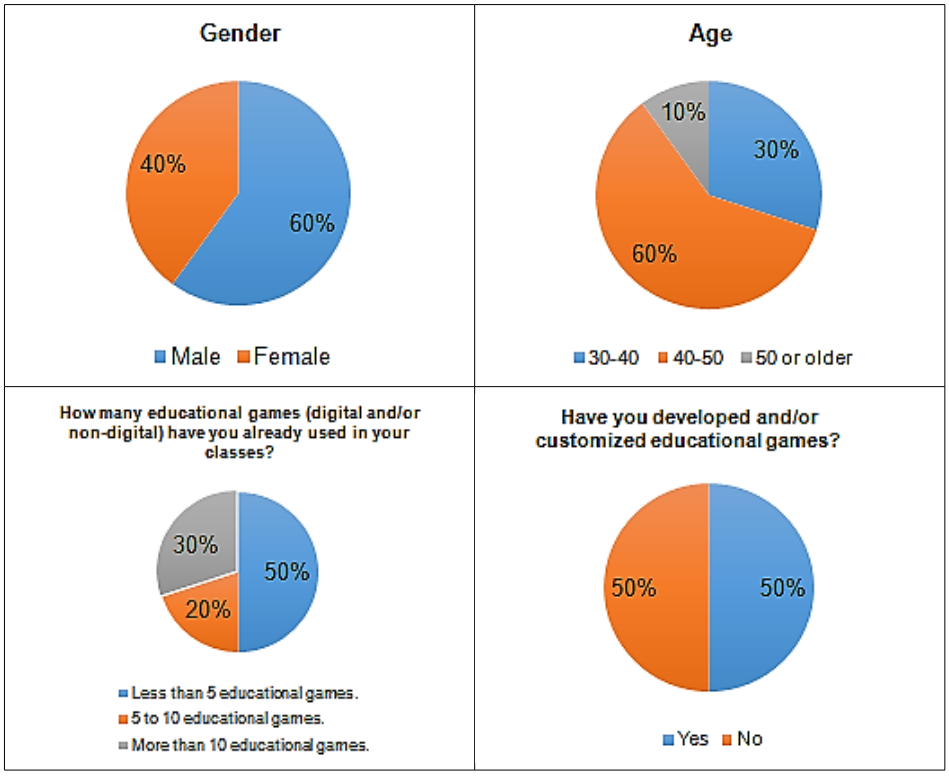

Fig. 5. Demographic data of the participants.

4.3. *Data Collection and Analysis*

Data collection was accomplished through the use of a specialized questionnaire, as already mentioned. The questionnaire used is based on the MEEGA+ educational game evaluation model designed by Petri *et al*. (2018). The MEEGA+ model is primarily used for the evaluation of EGs aimed at educating users in the field of Computer Science. In addition, it is worth noting that the questions were designed to be addressed to the teachers and not to the students.

More specifically, the MEEGA+ model assesses the quality of EGs by dividing the questions into two qualitative factors. The first quality factor relates to the *user's experience* and includes subcategories of questions that evaluate the player's *focused attention*, *fun*, *challenges*, *social interaction*, *satisfaction* and *confidence* during the game. It also includes questions about the *relevance* of the game's content with a specific field, its Perceived Learning and the protection it offers to the user in a case of an error. The second quality factor relates to the *usability* of the EG and the questions that are included assess its *learnability, operability, aesthetics* and *accessibility.*

Based on these, the questionnaire for the evaluation of BlocklyScript is divided into three sections:

- The first section includes six questions related to the participants' demographic data.
- The second section contains nine questions related to the usability of the game.
- The third section includes twenty-eight questions regarding the user's experience.

The questionnaire did not include questions from the MEEGA+ model on *social interaction* with other players since this feature is not supported by BlocklyScript. For the questions of the second and third section a 5-point Likert scale from -2 to 2 was used, where -2 equals "Strongly Disagree", -1 equals "Disagree", 0 equals "Indifferent", 1 equals "Agree" and 2 equals "Strongly Agree".

The data collected were then analyzed using Microsoft Excel. For the first section of the questionnaire referring to the participants' demographic data percentages were calculated, while for the second and third section, the median values and the percentages of each available response for each question were calculated as proposed by the creators of the MEEGA+ educational game evaluation model (Petri *et al*., 2018).

In order to estimate the reliability coefficient, or else the internal consistency, of the questionnaire the Cronbach's alpha was calculated. The reliability coefficient was .93798 which denotes a high internal consistency.

5. Results of the Pilot Evaluation

5.1. *Usability*

Fig. 6 presents the advantages of the game design, its ease of use and the support that it offers to the users in order to identify their mistakes inside the game. More specifically,

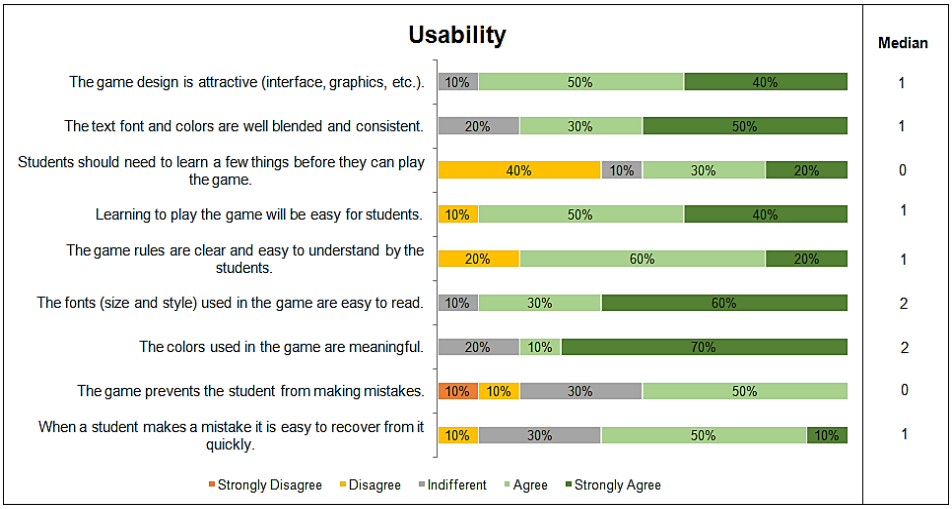

Fig. 6. Evaluation results on usability.

50% of the participants agree and 40% strongly agree that the game is attractive. The majority of the participants (30% agree and 50% strongly agree) agree with the choice of colors and consider them to be meaningful to the game (70% strongly agree).

In addition, 90% of the participants agree or strongly agree that students will easily learn how to handle the game. However, the results of whether students should know some basic things before playing the game are controversial, with 50% of the participants agreeing or strongly agreeing, 10% remaining neutral and 40% disagreeing. Therefore, this point can be improved by introducing more examples and levels of theory for smoother integration of students into the concepts of programming.

Furthermore, the game is considered to help the student identify possible mistakes and correct them relatively quickly with 50% of the participants agreeing and 10% strongly agreeing. But it is not sure if the game prevents students from making mistakes in designing their algorithms (median 0). Of course, the students learn through their mistakes and if they understand and correct them then it is more likely not to repeat them again in the future.

5.2. *Player Experience*

Fig. 7 presents the results of questions related to players' experience with BlocklyScript. The questions are grouped into categories according to their type. Following there is a commentary on the results of the player's experience ranging from the first category of questions (as shown in Fig. 7) to the last category.

Confidence. The first category of Fig. 7 presents the results of the educators' view on the confidence that students will have in the game. All participants agree or strongly agree

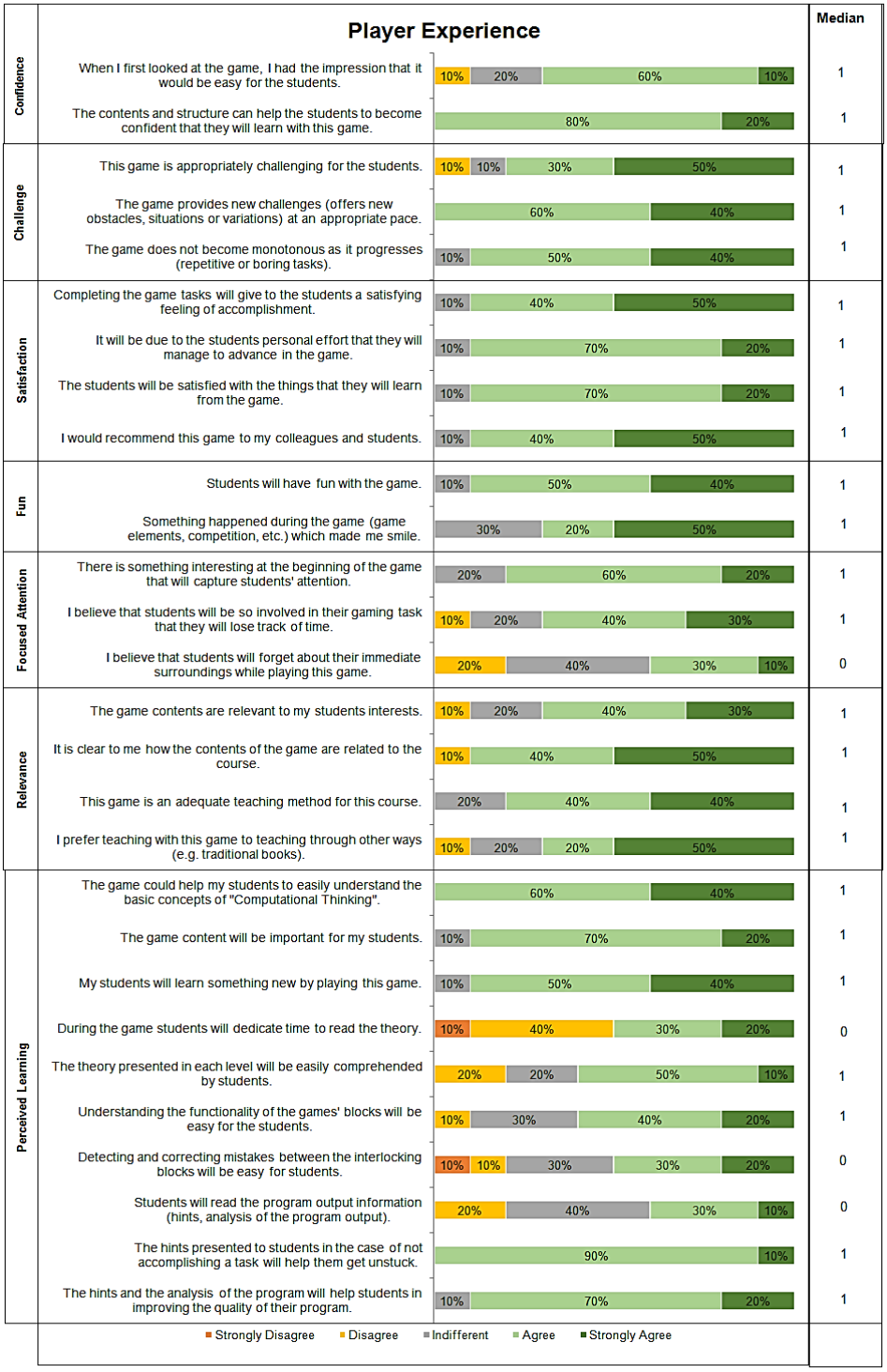

Fig. 7. Evaluation results on player's experience.

that with the structure and educational material of the game, students will feel confident that they will learn by playing. Most of the participants also believe that BlocklyScript will be easy for students (median 1).

Challenge. The challenges that the student will face while playing are numerous, thus keeping his/her interest undiminished. The second category of Fig. 7 shows the results of the questions regarding the challenges that the game offers. All the questions managed to get a median of 1 (agree) and it is worth mentioning that teachers do not believe that the game becomes monotonous as it evolves and there are no elements to make it boring. Students will thus be motivated to continue playing by unlocking new levels and learning new programming concepts.

Satisfaction. The vast majority (90%) of the teachers (strongly) agrees that students with their own personal effort will be able to unlock all levels of the game and be satisfied with their achievements, as well as with the new programming concepts that they will learn. Finally, 90% of the participants would recommend this game to colleagues and students.

Fun. The participants agree (median 1) that students will be entertained by the user friendly interface, the cheerful tone of music and the interesting script.

Focused attention. The fifth category of questions refers to students' focused attention. Participants agree (median 1) that the good game design will capture the student's attention at startup and thus the student will lose track of time. However, opinions contradict (median 0) on whether the game will be able to absorb the player to such an extent that s/he will forget his/her surrounding as s/he plays.

Relevance. The sixth category presents the results of the evaluation concerning the relevance of the educational material to the IT field. According to Fig. 7, the content of the game is related to programming (40% agree, 50% strongly agree) and students' interests in school (40% agree, 30% strongly agree). In addition the teachers argue that BlocklyScript is an adequate method of learning the basic concepts of programming (80% agree or strongly agree) and that they would prefer to use this game versus a textbook to teach their students (median 1).

Perceived Learning. The last category of Fig. 7 presents the questions related to the expected learning outcomes. All the participants agree or strongly agree (60% agree, 40% strongly agree) that the game will help students to comprehend easily the basic concepts of CT. In addition, 90% believe that students will learn new programming concepts that will be useful enough to develop their CT. However, there is some doubt (median 0) as to whether the students will devote the necessary time to read the theory and the area of debugging and to easily identify the errors between the interconnected blocks. Finally, the results regarding the usefulness of the help tips and advices contained in the debugging area are positive. Specifically, over 90% of the participants agree or strongly agree that the tips provided in the debugging area will help students to understand their mistakes, recover from them and improve the quality of their algorithm.

6. Limitations

The main goal of the presented study was to investigate teachers' perceptions on the usability and user experience of the proposed EG targeted to teaching and learning programming and CT to young students. This was deemed necessary in order to validate the design of BlocklyScript and making the necessary revisions prior its usage by young students. Although the teachers that play-tested BlocklyScript and evaluated it had all experience in utilizing EGs in the classroom and moreover half of them on designing and/or customizing existing games, their number (ten participants) was small and this is a limitation of the study. However, their positive evaluation and the feedback provided us more confidence in making available the game to interested teachers and researchers for carrying out a study on utilizing the game with young students. Such a study is necessary for drawing safe conclusions about the potential educational value of BlocklyScript and the validation of the design decisions taken.

7. Discussion

Regarding *the teachers' perceptions on the usability and user experience of BlocklyScript* (research question), the results of the pilot evaluation by ten IT teachers of primary and secondary schools were promising. Specifically, the questions related to *confidence*, *challenge*, *satisfaction*, *fun* and *relevance* of the EG captured teachers' positive attention (median 1 – agree). In addition the questions related to *usability* indicate that the game design is attractive, colors are meaningful and it is expected to support students' progress by providing useful tips when getting stuck. Finally, based on teachers' perceptions, students are expected to easily understand the basic concepts of CT (60% agree, 40% strongly agree) by playing BlocklyScript which is the primary aim of this game. Similar favorable results were indicated in a research for the effectiveness of MiniColon for cultivating CT and programming skills to children (Ayman *et al*., 2018).

However there are some aspects of the EG that raised doubts between the teachers. Those grey areas (median 0 – indifferent) of the evaluation combined with the teachers' open-ended comments could be taken under consideration to improve the overall quality of BlocklyScript as an educational tool. It's worth mentioning that two of the comments have been implemented right after the evaluation of the first version of the game. The first comment noted that the game should be translated to Greek (authors' native language) to help young students comprehend with ease the basic programming concepts. The second comment was affiliated with the redesigning of the game's last level in order to provide the user with a feeling of achievement and associate it with the final goal of the game's scenario.

In addition a teacher suggested adding hints at every step of the designing phase of the algorithms at each level instead of running and resetting the game, which currently results in losing some points. This comment is associated with the uncertainty of the teachers about the capability of the students to detect and correct mistakes between the

interlocking blocks (median 0). A step by step execution of the designed algorithms could also be implemented in a future version to allow users understand exactly the block that is being executed at a certain time and help them identify their mistakes (Giannakoulas & Xinogalos, 2019). *Explanatory visualization*, that is explanatory messages in natural language for each step during execution could further support students in comprehending the semantics of the statements and debugging their programs (Xinogalos *et al*, 2006).

Furthermore teachers were concerned whether young students will dedicate time to read the theory and the program output information (hints, analysis of the program output). By using pop out messages and introductory short length videos the EG could draw students' attention and diminish the time spent on reading. This approach has been used by most of the games – specifically Minerva, Code Combat, Hour of Code: Minecraft, and Rapid Router – presented in section 2 (Danks *et al*., 2019; Du *et al*., 2016, 2018; Lindberg & Laine, 2018). "Hour of Code: Minecraft" also includes introductory videos from celebrities, such as Bill Gates, in order to inspire the students and highlight the importance of coding in their lives.

Finally teachers argue whether the game could immerse students to a point of forgetting their immediate surroundings. "*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*" (Giannakoulas and Xinogalos, 2018, p. 21). For this reason more time could be devoted to design appealing graphics and select better sound effects and background music, as well as enhancing the game mechanics.

8. Conclusion

In this article the rationale of the design, as well as the results of a pilot evaluation of the EG BlocklyScript that aims to teach programming concepts and cultivate CT skills to young students was presented. The integration of visual programming with the help of Blockly in the game intends to make it easier for players to learn how to use the game and to understand the concepts presented per level more easily. By using the blocks, users can directly apply the theory they read and collect starts, defeat aliens and avoid obstacles. So at every level of the game, users face new challenges and are expected to keep their interest undiminished. In contrast with most of the existing games of this kind, students do not have just to guide their avatar to follow a path for reaching a specific point, but they have to fight with enemies, avoid obstacles, recognize patterns and guide their avatar efficiently through the platforms in each level in order to provide the optimum solution and be rewarded.

Based on the results of the pilot evaluation by experienced IT teachers, BlocklyScript is considered to be a reliable tool for developing CT skills to young students. Moreover, based on the results of the pilot study the main design decisions that were guided by the model proposed by Ibrahim and Jaafar (2009) seem to be validated. Although the number of participants was small some interesting pedagogical implica-

tions were recorded. The three virtues of EG reported by Rebah (2019), which are the intrinsic motivation of learners, situated learning and learning from mistakes seem to be valid for BlocklyScript that is expected to raise students' confidence and satisfaction, entertain them through challenging tasks and motivate them to learn through playing. However, it must be noted that extra care should be taken in the case that the game will be used for self-learning outside of the classroom. Although BlocklyScript incorporates the necessary theory, as well as an analysis of a program's output and hints for correcting or even improving its quality, the participants raised doubts as to whether young students will devote time in studying and comprehending this information. So, if this or similar games are going to be used for self-learning, the educational content should probably be presented in a more interactive way (for example through video tutorials, some kind of animation or pop-up messages) in order to attract students' attention. In the case of using the EG inside the classroom the teacher could talk with the students regarding the educational material presented at the beginning of each level of the game in order to make sure that they have comprehended the underlying programming and/or CT concepts. This should be followed by a debriefing session (Crookall, 2010) in class – after trying out to solve each level – in order to comment on the feedback provided during the debugging process, assist students in using this feedback for correcting their programs and ultimately helping them to adopt a systematic debugging strategy.

Of course, in order to draw safe conclusions regarding the validity of the aforementioned pedagogical implications and the quality of the game, it has to be evaluated by a larger group of teachers and by the students themselves and also its educational value has to be thoroughly investigated. Finally, based on this study some interesting implications to practitioners and researchers, as well as suggestions for further research are presented in the following sections.

9. Implications to Researchers and Practitioners

EGs are an alternative way of educating people and especially young students. The main purpose of each EG is to present to the user some concepts related to a particular scientific field in a simple and entertaining way. Moreover the concept of "play" is familiar to young students and its combination with education is more attractive to their interest than a traditional book. In addition, EGs enhance users' self-learning by providing all the necessary tools that they will need in order to learn how to use it and comprehend the theory of a particular scientific field.

The fact that EGs promote self-learning provides great opportunities for educating students that cannot attend school due to various reasons, such as the extraordinary situation that the world is facing due to the COVID-19 pandemic. In such cases, distance learning either synchronously or asynchronously is the only solution for keeping students in touch with their school, teachers and schoolmates that is necessary not only for their education but their well-being as well. UNESCO (2020) has highlighted the importance of distance learning in the era of COVID-19.

A game like BlocklyScript that supports the creation of student accounts gives the chance to students to learn programming concepts and acquire CT skills in a playful way, at their own pace from the leisure of their homes. Moreover, storing students' achievements and difficulties during game-play in a data base, provides teachers with a clear picture of students' progress. Data such as tasks accomplished, quality of solution, hints retrieved and problems encountered are invaluable tools at the hands of the teacher in order to support students in learning programming (Malliarakis *et al*., 2014b). It is important for *teachers to invest time for getting familiar with and utilizing appropriate EGs for supporting the teaching and learning of programming* not only in the extraordinary situation of COVID-19 but also for in classroom use and home assignments as proposed by Giannakoulas and Xinogalos (2018). Moreover, it would be interesting for researchers to: *comparatively analyze existing EGs for their readiness in supporting the distance education of programming and making informed suggestions to the community; studying the effects of utilizing a game like BlocklyScript in a distance learning situation and providing guidelines for a successful usage out of the typical classroom*. There are not many empirical studies on the effects of EGs on teaching and learning programming to and by young students in general (Giannakoulas and Xinogalos, 2020), while such studies in a distance learning mode do not exist to the best of our knowledge.

When it comes to designing EGs for programming and CT, our experience has shown that utilizing a specialized EG design framework, such as the one proposed by Ibrahim and Jaafar (2009), can provide guidance in effectively designing the game especially in the case that it is not designed by a big team consisting of experts on education, storytelling, game design and graphics. Moreover, a pilot evaluation of the game based on play-testing by experienced teachers and a survey based on an established model, such as the MEEGA+ model (Petri *et al*., 2018), can provide insights on the quality of the game.

10. Suggestions for Future Research

The suggestions for future research are several. The most important ones include a study with young students as mentioned in the section on the limitations of the current study. Moreover, it would be interesting to investigate the effectiveness of BlocklyScript and other similar EGs on a distance education situation. Currently, the COVID-19 pandemic demanded moving from face-to-face teaching to a distance education model. Although, higher education institutions were more ready for supporting distance learning, primary and secondary schools faced a great challenge. Although EGs support self-learning, utilizing them as distance education tools for teaching programming to young students has to be thoroughly investigated in order to provide best practices.

Regarding the design and the features of BlocklyScript several extensions could be made based on the results of its pilot evaluation. The most important ones were presented in the Discussion section, while its usage in distance education settings would demand even more extensions, such as support for learning analytics and an enhanced dashboard for the teacher in order to monitor students' progress (Giannakoulas and Xinogalos, 2020; Malliarakis *et al*., 2014b).

References

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75B, 661–670. DOI: 10.1016/j.robot.2015.10.008

Ayman R., Sharaf N., Ahmed G., Abdennadher S. (2018). MiniColon; Teaching Kids Computational Thinking Using an Interactive Serious Game. In: Göbel S. *et al*. (eds) Serious Games. JCSG 2018. *Lecture Notes in Computer Science*, vol 11243, 79–90. DOI: 10.1007/978-3-030-02762-9_9

Bell, J., & Bell, T. (2018). Integrating Computational Thinking with a Music Education Context. *Informatics in Education*, 17(2), 151–166.

Combéfis, S., Beresnevičius, G., Dagienė, V. (2016). Learning programming through games and contests: overview, characterization and discussion. *Olympiads Informat*. 10(1), 39–60.

Crookall, D. (2010). Serious Games, Debriefing, and Simulation/Gaming as a Discipline. *Simulation & Gaming*, 41(6), 898–920. DOI: 10.1177/1046878110390784

Dagienė, V., Stupurienė, G. (2016). Bebras – A Sustainable Community Building Model for the Concept Based Learning of Informatics and Computational Thinking. *Informatics in education*, 15(1), 25–44.

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`

Du, J., Wimmer, H., Rada, R. (2016). "Hour of Code": Can It Change Students' Attitudes toward Programming? *Journal of Information Technology Education: Innovations in Practice*. 15. 53–73. DOI: 10.28945/3421.

Du, J., Wimmer, H., Rada, R. (2018). "Hour of Code": A Case Study. *Information Systems Education Journal,* 16(1), 51–60. `http://isedj.org/2018-16/`

Fessakis, G., Prantsoudi, S. (2019). Computer Science Teachers' Perceptions, Beliefs and Attitudes on Computational Thinking in Greece. *Informatics in Education*, 18(2), 227–258.

Giannakoulas A., Xinogalos S. (2019) Current Trends in On-line Games for Teaching Programming Concepts to Primary School Students. In: Tsitouridou M., A. Diniz J., Mikropoulos T. (eds) Technology and Innovation in Learning, Teaching and Education. TECH-EDU 2018. *Communications in Computer and Information Science*, vol 993. Springer, Cham. DOI: 10.1007/978-3-030-20954-4_5

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. `http://doi:10.4018/978-1-7998-4576-8.ch001`

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. *Educ Inf Technol*, **23,** 2029–2052. DOI: 10.1007/s10639-018-9702-x

Ibrahim, R., & Jaafar, A. (2009). Educational games (EG) design framework: Combination of game design, pedagogy and content modeling. In *International Conference on Electrical Engineering and Informatics, 2009. ICEEI'09* (Vol. 1, pp. 293–298). IEEE. DOI: 10.1109/ICEEI.2009.5254771

Jakoš, F., Verber, D. (2017). Learning Basic Programing Skills With Educational Games: A Case of Primary Schools in Slovenia. *Journal of Educational Computing Research*, *55*(5), 673–698. DOI: 10.1177/0735633116680219

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

Lindberg, R., Laine, T. (2018). Formative evaluation of an adaptive game for engaging learners of programming concepts in K-12. *International Journal of Serious Games*, 5, 3–24. DOI: 10.17083/ijsg.v5i2.220

Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2014a). Designing educational games for computer programming: A holistic framework. *Electronic Journal of e-Learning*, 12, 281–298.

Malliarakis C., Satratzemi M., & Xinogalos S. (2014b). Integrating Learning Analytics in an Educational MMORPG for Computer Programming. *2014 IEEE 14th International Conference on Advanced Learning Technologies*, Athens, pp. 233–237, DOI: 10.1109/ICALT.2014.74

Petri, G. & Gresse von Wangenheim, C. & Borgatto, A. (2018). MEEGA+: A Method for the Evaluation of Educational Games for Computing Education (Technical report, INCoD/GQS.05.2018.E). Brazilian Institute for Digital Convergence.

Pinelle, D. & Wong, N. (2008). Heuristic evaluation for games: Usability principles for video game design. *Conference on Human Factors in Computing Systems – Proceedings*. DOI: 10.1145/1357054.1357282

Prensky, M. (2003). Digital game-based learning. *Computers in Entertainment (CIE)*, 1(1), 21–21.

Rebah, H. (2019). The educational effectiveness of serious games. *Médiations & médiatisations,* 19(2), 131–155.

Saito, D., Washizaki, H., Fukazawa, Y. (2017). "Comparison of Text-Based and Visual-Based Programming Input Methods for First-Time Learners", J*ournal of Information Technology Education*: Research*,* 16, 209–226. DOI: 10.28945/3775

Stege, L., Van Lankveld, G., & Spronck, P. (2011). Serious games in education. *International Journal of Computer Science in Sport*, 10(1), 1–9.

Svinicki, M.D. (1999). New directions in learning and motivation. *New Directions for Teaching and Learning*, 80, 5–27.

Tonbuloğlu, B., & Tonbuloğlu, I. (2019). The Effect of Unplugged Coding Activities on Computational Thinking Skills of Middle School Students. *Informatics in Education*, 18(2).

UNESCO (2020, June 22). *Education: From disruption to recovery.* Retrieved from: `https://en.unesco.org/covid19/educationresponse/`

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.

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

Zaharija, G., Mladenović, S., Boljat, I. (2013). Introducing basic programming concepts to elementary school children. *Procedia-Soc. Behav. Sci*. 106, 1576–1584

Zyda, M. (2005). From visual simulation to virtual reality to games. *Computer*, 38(9), 25–32.

C. Karakasis holds a Bachelor and Master degree since 2020 from the Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece. His interests include programming and educational games. The educational game BlocklyScript was designed and implemented in the context of his master thesis.

S. Xinogalos is an associate professor at the Department of Applied Informatics at the University of Macedonia, Thessaloniki, Greece. He is a member of the Software and Data Engineering Lab and the Educational Technology Research Group at the University of Macedonia. His research interests include Programming Environments and Techniques, Object-Oriented Design and Programming, Didactics of Programming, Educational Environments and Games for Programming, and Serious Games. He has published more than 100 peer-reviewed articles in international journals, conference proceedings and book chapters.