SUPER MARIO: A COLLABORATIVE GAME FOR THE LEARNING OF VARIABLES IN PROGRAMMING

Christos Theodorou[1], and Maria Kordaki[2]

ABSTRACT

This paper presents the design and implementation of a collaborative game for learning about the various uses of variables in programming, by high school students. This is a 4-level game that was designed and implemented using the programming environment Scratch integrating the well known collaborative method Jigsaw (1). The basic learning goals of this game are to provide students with opportunities: a) to conceptualize some different aspects of variables in programming, by playing, interacting, and modifying this game and b) to become equipped with appropriate knowledge -at the end of the proposed activities- to collaborate and construct their own games themselves. Students are divided into 4 expert groups, by applying the aforementioned collaborative method, where each team explores one specific level of the game, which focuses on learning about a specific category of variables in programming. Afterwards, the students from the expert groups try to design their own games and then return back to the original Jigsaw groups in order to teach their colleagues -through their involvement in a sequence of game based activities-about each of the different categories of variables. In this article the aforementioned levels of this game and the specific learning activities to be assigned to these groups are presented.

Key words: variable, programming, computer games, Scratch

1. INTRODUCTION

Games are viewed as being the most ancient and time-honored vehicle for education (2) and are also among the most enjoyable and motivating activities for the young (3). Specifically, games provide a meaningful context for students because the outcome,

[1] Mr Christos Theodorou is a prospective Computer Engineer in the Dept of Computer Engineering and Informatics, Patras, University, Greece. His interests include Educational Technology and especially, design and implementation of Computer games.

[2] Mrs Maria Kordaki holds a bachelor in Mathematics, a diploma in Engineering, a master in Education and a PhD in Educational Technology. She is assistant Professor in the Dept of Cultural Technology and Communications, University of the Aegean, Greece. Her research interests include Educational Technology, game based learning, digital storytelling and the use of technology in Didactics of Mathematics and Computer Science. She is author of more than 125 scientific publications in international refereed journals, scientific books as well as international and national conferences.

winning or losing, can be important to them (4). This motivation to succeed stimulates students to learn the concepts in focus by connecting with previous experiences, both inside and outside the classroom (5). Moreover, research into games and play has demonstrated that players can attain a state of ‘flow’ (6) summarized as “the state in which we are so involved in something that nothing else matters”. In fact, the game-based learning approach brings with it the involvement and the excitement of accomplishment (7). To this end, the role of engaging learners in meaningful and enjoyable learning activities is acknowledged as crucial for the learning of any subject (8-9), let alone the learning of fundamental Computer Science concepts and skills (10).

Leading scholars have also long argued that computer games provide a compelling context for children’s learning (11). In the course of playing appropriately-designed computer games, children can be introduced to new concepts, topics and skills which they can continue to explore through offline reading, discussions or activities (12). In addition, well-designed computer games can serve as useful tools for both formal (i.e., classroom) and informal (i.e., outside class) education (12). To this end, it is worth noting that computer games play a central role in young people’s lives (11-14) and constitute their most popular computer activity at home (15).

Appropriately-designed educational computer games can become powerful learning environments because they can support student learning in terms of (14): (a) multi-sensory, active, experimental learning, (b) activation and use of prior-knowledge in order to advance, (b) self-correction by providing immediate feedback on student actions, (c) self assessment by exploiting the scoring mechanisms, (d) acquisition of essential learning ‘competencies’ such as logical and critical thinking as well as problem-solving skills (16), and (e) learning in different ways from those often in evidence, or explicitly valued, in school settings (13). In fact, young people seem to expect different approaches to learning, shaped by their frequent interaction with computer games outside schools (17), and (f) motivation and computer-based interaction.

The motivational effectiveness of computer games has been supported by the findings of various empirical studies. In particular, in the TEEM (‘Teachers Evaluating Educational Multimedia’) report (16), teachers and parents recognized that the playing of computer games can support valuable skill development such as strategic thinking, planning and communication, application of numbers, negotiating skills, group decision-making and data-handling. Yet, despite this, the educational effectiveness of computer games aiming at concrete learning objectives for specific learning subjects included in school curricula is still under-researched (18). However, computer games have been effective in raising achievement levels of both children and adults in various areas of knowledge such as Science and Math, Language and Computer Science, where specific learning objectives can easily be stated (18-19).

Research in e-learning also points out that involving learners in online collaborative learning activities could provide them with essential opportunities, such as: motivation for active engagement in their learning (20), to extend and deepen their learning experiences, to try new ideas and improve their learning outcomes (21), to trigger their cognitive processes (22), to enhance their diversity in terms of the learning concepts in question (23) as well as to interact socially and develop a sense of community and of belonging online (24). On the whole, computer-supported collaborative learning has been recognized as an emerging paradigm of educational technology (25). Despite this, many teachers remain unsure of why, when, and how to integrate collaboration into their teaching practices in general as well as into their online classes (26). To encourage teams to achieve effective collaboration some amount of structuring may be necessary (27). One way to structure collaboration is through the use of computer-supported collaborative design patterns. A pattern is seen as something that will not be reused directly but can assist the informed teacher to build up their own range of tasks, tools or materials that can draw on a collected body of experience (28). Among the various collaborative design patterns the JIGSAW (1) collaborative pattern is of special interest.

The teaching of programming is one of the main challenges in Computer Science, despite it being not only an essential topic proposed for a K-12 curriculum, and a fundamental subject in studying Computing at Tertiary level, but also a ‘mental tool’ of general interest (29) where problem-solving skills can be encouraged in learners. In truth, programming is more a mental skill than a body of knowledge (30). It is a complex task, including understanding of the task at hand, method finding, coding, testing and debugging of the resulting program (31). It is essential to note here that students encounter serious difficulties in performing any or all of the above (30-34). It is also worth noting that, there has been strong interest in the adoption of effective methods to improve the ability of students to comprehend and solve computational problems requiring programming solutions (33-36). As regards the concept of variable in programming students also seemed to illuminate serious difficulties (37). In fact, students seemed to present difficulties in the understanding of the dynamic character of a variable in programming and confuse it with its meaning and representation in mathematics. The initialization, updating and testing of variables are also difficult for students, especially when they encountered inside loops and conditions (38).

To this end, Scratch (www.scratch.mit.edu) is a new programming language and also, an environment for the learning of programming. Specifically, Scratch is based on the same concepts as other typical programming languages. However, Scratch does not support the learning of programming through writing long programs in text format using complex structures and following rigid syntactical rules which are mainly familiar to expert programmers (39). In fact, the Scratch environment motivates beginners in programming to be involved in programming by exploiting their interest in games and play. In the environment of Scratch, students can construct intuitively and easily (drag and drop) their own games at the same time receiving visual feedback for their actions for self correction. Scratch is easily accessible to everyone, as it is free software that is supported by an international community of users. Members of this community exchange not only technical information but also complete learning activities. Despite the above, a collaborative and structured, learning activity using the JIGSAW method within Scratch have not yet been reported.

Based on all the above, a 4-level computer game (named: Super Mario collaborative game) was constructed using the tools provided by Scratch while at the same time exploiting the JIGSAW collaborative method. The aim of this game is to serve as a possible learning environment for the concept of variable in programming by high school students. Each of these levels focus on the learning of different aspects of variables in programming, namely: (a) variable as a structure in object-oriented programming, describing basic characteristics of an object, eg. position, costume, size, (b) variables as they involved in equations/ functions used in Mathematics and Physics, eg. equations of circle, sinus and velocity, (c) as general purpose variables in procedural programming eg. Register, counter, pointer, table, que, and (d) as signal for communication between processes and objects and synchronization. Such game integrating the JIGSAW collaborative method within Scratch for the learning of the notion of variable in programming has not yet been reported.

In the following section of this paper, a brief description of the Jigsaw collaboration method is presented. Then, the design and the implementation of the Super Mario collaborative game are reported. Finally, the design of this game is discussed and conclusions and future research plans are drawn.

2. THE JIGSAW COLLABORATIVE METHOD

The *Jigsaw* method was originally proposed by E. Aronson (1) at the University of Texas and the University of California. Hundreds of schools have Jigsaw-based activities in their classrooms with much success (see http://www.jigsaw.org). Jigsaw has been seen as a method that can support both cooperative learning and collaborative situations. Gallardo (40) also thought that this method could be well situated within the constructivist framework of learning. In addition, many researchers have proposed the implementation of this method within the online context (40-41), despite the fact that Jigsaw was originally proposed for face-to-face education (1).

Specifically, the Jigsaw method is a cooperative/collaborative learning strategy which could enhance the process of listening, commitment to the team, interdependence and team work. Each member of the team has to excel in a well-defined subpart of the educational material, undertaking the role of expert. The experts form a different group to discuss the nuances of the subject and later return to their teams to teach their colleagues. The ideal size of teams is 4 to 6 members. Specifically, the implementation of the Jigsaw method could be realized through the following process: 1) Divide the problem into sub-problems, 5) Create heterogeneous groups, 3) Assign roles and material to each student, 4) Form group of experts, 5) Let experts study the material and plan how to teach their colleagues, 6) Let experts teach in their groups, 7) Assess students.

Through Jigsaw, the following goals could be achieved: (a) Building of interpersonal and interactive skills, (b) Ensuring that learning revolves around interaction with peers, (c) Holding students accountable among their peers, and (d) Encouraging active student participation in the learning process.

3. THE SUPER MARIO COLLABORATIVE GAME

Initially the students, having formed the original Jigsaw groups, set up four expert groups; each of them has to deal with one level of the said game. Students in each group will first play the corresponding level. Then, each team is assigned with a number of activities in order to understand the category of variables, on which the corresponding

level focuses, namely: (a) execution of a specific scenario or group of scenarios of the program and notice the changes on the screen. This way, students can easily correlate these changes with the changes of the relevant variables, (b) modification, removal or add scenarios, to understand their function. It is worth mentioning that the functionality of each object is described by some scenarios that run in parallel and each one has been designed to perform a specific and clear sub-function of the object. Therefore, students need to discover, through these activities, these sub-functions, (c) construction of programs – with given (or not), 'scenes' and 'objects' - that implement specific functions, using the kind of variables, on which the respective level of Mario game focuses. These programs can be used by these experts to teach their classmates, when they return to their original Jigsaw groups.
In the end of the game play activity, each student from each original Jigsaw group will be evaluated. This evaluation will be based on the implementation of a series of programming activities regarding the use of all type of variables they have met in all levels of the Mario game. In addition, students in the original groups have to work together to integrate all levels into a single one. Below, the levels of this game are described in further detail.

A. Expert group – level 1

Mario moves using the arrow keys and must across two scenes, where the 1st is shown in Figure 1a. He has to take a mushroom and finish the level, having avoided both the sea lanes and the shots that the cloud throws. This level focuses on the use of structural variables (size, costume, position and background). These variables are part of the Scratch environment and are related with the basic properties of the objects used. In fact, these variables are: 1) *Size* of Mario, that is initialized with a value, which is decreased when a shot touches Mario, resulting his automatic miniaturization, 2) *Costume* of Mario (with 7 possible values). 3) *x position & y position* of Mario, used in order to examine if Mario touches the sea surface in the background demonstrated in Figure 1, or if he is a the end of it. This variable is also used to change the sprite's position depending on the arrow key pressed. In addition, this variable is used to direct the shoot to Mario's current position, and 4) *Background*, that is a property of the object "stage" which takes 3 different values.

To help students understand the concept of structural variables, a series of programming activities are suggested to this expert group, like: a) execution of the Mario script that is shown in Figure 1b and observation of both; Mario figure and the structural variable «costume», before and after the mushroom touches it, b) having removed the Mario' s scenario which implements Mario's motion on axis y, we can ask students to start playing the level, observe the problem (Mario cannot jump) and correct it by adding the appropriate scenario, with the appropriate position-y initialization, c) construct a program to teach their classmates in the original Jigsaw groups. To this end, some ready-made objects are suggested to these experts for use. In this program, a 'sprite' «boy» walks left/right using the arrow keys of keyboard and passes from three different landscapes, where its size is different from place to place (see some indicating Figures 1c, 1d). The structural variables used here are: 1) *Background* of the stage, 2) *Size* of the boy, 3) *Position* (x, y) of the boy, and 4) *Costume* of the boy (5 different costumes should be given). Certainly, students can also build their own game, using objects of their choice.

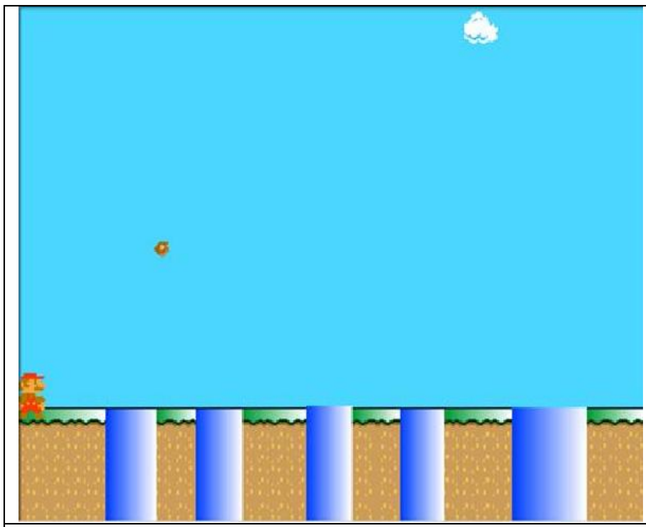


Fig. 1a. *Snapshot of level 1*

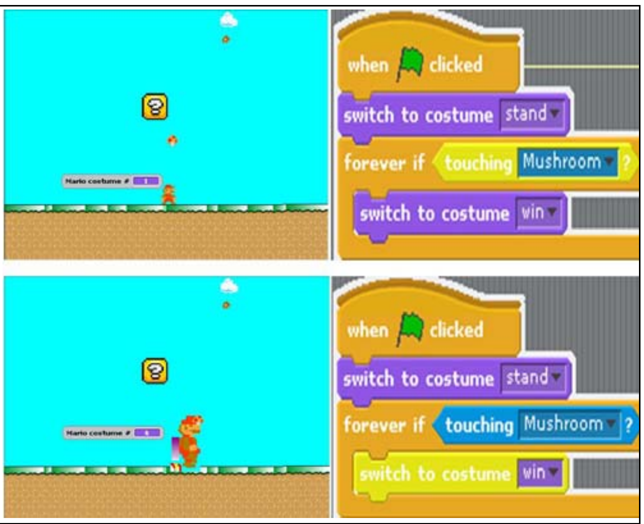

```
when [flag] clicked
switch to costume stand
forever if touching Mushroom ?
    switch to costume win
```
```
when [flag] clicked
switch to costume stand
forever if touching Mushroom ?
    switch to costume win
```

Fig. 1b. *Snapshot illustrating the relation between the variable costume and the appearance of Mario.*

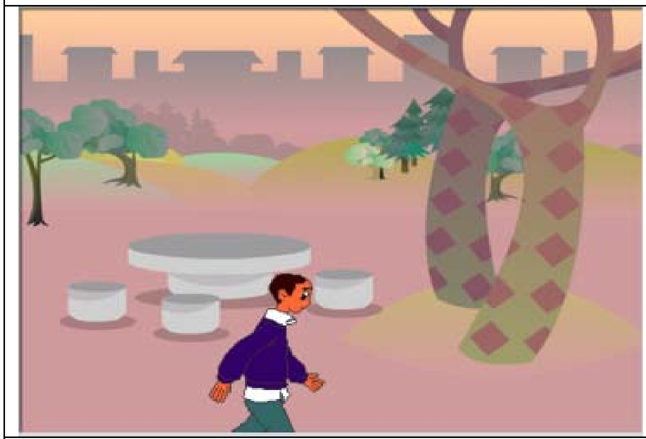


Fig. 1c. *Scene 1 of the example proposed to be constructed by expert group - level 1*

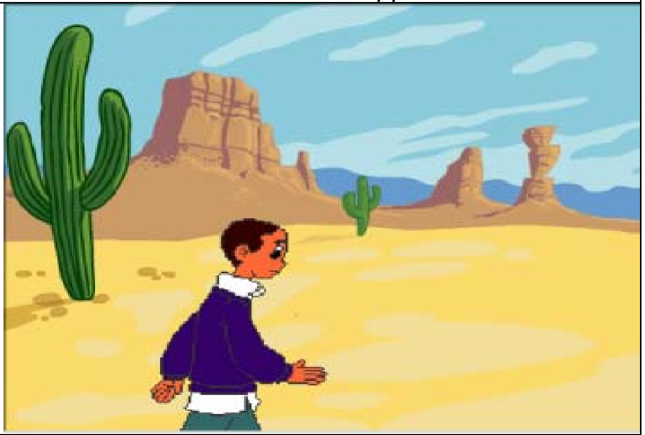


Fig. 1d. *Scene 2 of the example proposed to be constructed by expert group- level 1*

B. Expert group – level 2

Here, Mario can fly using the arrow keys. The goal is to reach to the yellow arrow that is placed on the opposite side, avoiding both various flying objects and the sea (Figure 2a). In this level, variables for the description of equations are implemented.

As an example, the ‘flame’ object, which is thrown by the ‘dragon’ object, makes a sinusoidal motion described from the following function: Y= A*sinX+Yo. For the description of this function, the following variables were created: 1) constant A for the width of this motion, 2) constant Yo for the initial height, 3) variable X, as the domain field of the sinusoidal function, 4) variable Y, as the range field of the sinusoidal function. It should be also noted, that the equation of motion for each of the flying objects is implemented by 3 ‘scripts’: One describing the equation of motion, one that gives values to the domain field and one that assigns the values of the domain field and the range field to the structural variables «Position x», «Position y», correspondingly.

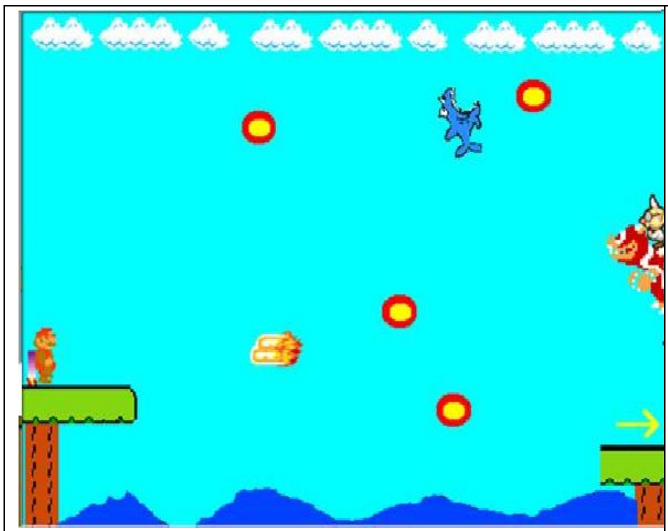


Fig. 2a. *Snapshot of level 2*

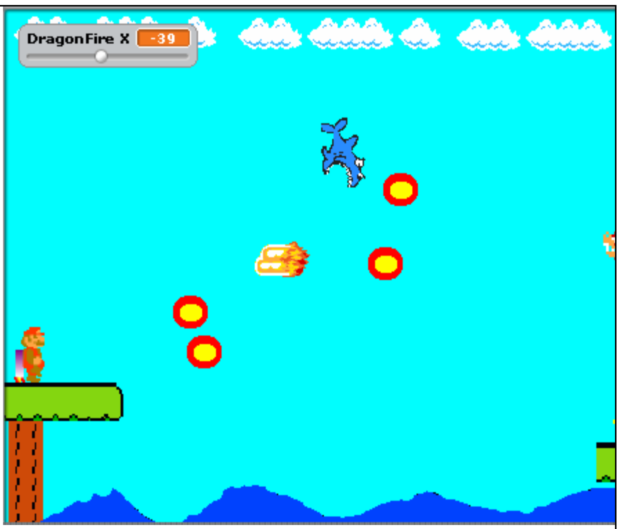


Fig. 2b. *Handling the axis-x motion of the object flame directly by the user.*

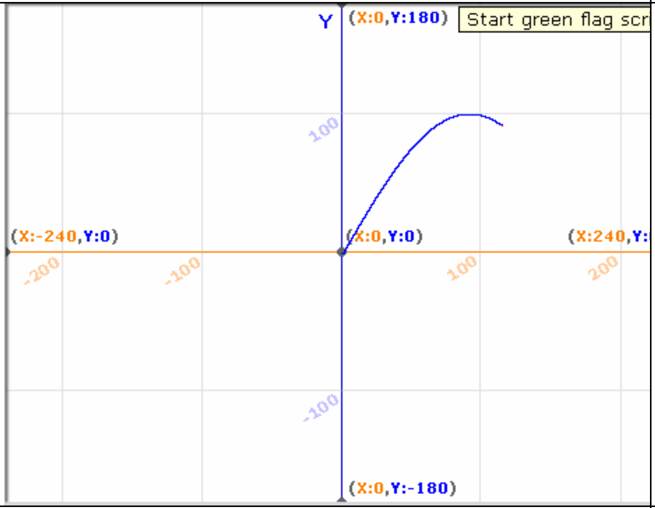


Fig. 2c. *Snapshot 2 of level 2 teaching example*

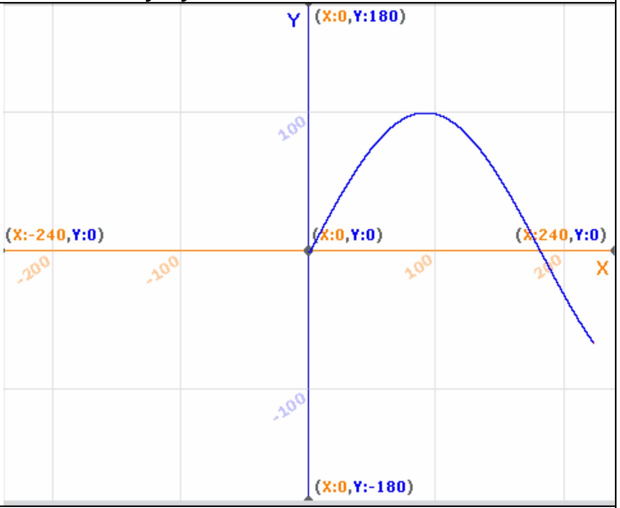


Fig. 2d. *Snapshot 4 of level 2 teaching example*

To help students of this expert group to understand the use of variables that participate in these functions some programming activities have also been prepared. Some examples of such activities are: (a) Run all ‘scripts’ of the object ‘flame’ and observe the motion of this object, while variable X is dynamically changing using the mouse (Figure 2b). Find the script which automatically performs the job that is manually performed, (b) Modify of the appropriate ‘Flame’s ‘script’, in order to execute the function y=2*x+5, after it has crossed half of the screen, (b) Construct a program in order to help their classmates in original groups to learn about the type of variables that this level focuses on. For example, given a coordinate system x-y: the program under construction will consist of the system x-y, as the background, and a ‘point’ object that will draw the function y=100*sin(x) (see some indicating Figures 2c, 2d).

C. Expert group – level 3

Figure 3a shows a snapshot of level 3. Mario moves left and right by the corresponding arrow keys, trying – within a certain time and having a specific number of lives – to beat the dragon and enter the castle (level 4). For this purpose, Mario has to gather bunches of bananas that fall from the sky. For each bunch of bananas has an amount of energy, different from bunch to bunch. Once Mario collects five bunches, he selects the bunch with the most energy and tries to attack against the dragon, decreasing the dragon’s energy by an equivalent amount. Two types of general purpose variables are implemented in this level: 1) *Local*: they refer to a specific object and are visible from this object only. For example, the list «basket», of length 5, represents Mario’s basket, which gathers the energy of each bunch of bananas and is implemented as a LIFO queue. 2) *Global*: they don’t belong to a specific object but are instead visible and manageable from all objects. For example, the variable «*State*» determines the phase that level 3 is in. In fact, this level has 4 phases, thus, variable *State* can have 4 different values: *Intro, Fight, Win, Lose*.

For the understanding of general purpose variables some programming activities are suggested to the students of this expert group. Examples of such activities are: (a) execute the Mario’s script that handles the following case: Mario’s basket is filled with 5 values, thus, the greatest of them is to be found, to be positioned at the top of the list and afterwards, Mario is to attack the dragon, decreasing his energy by this top value. Thus, students should be asked to identify and explain the role of the variables that are involved in this case. Figure 3b illustrates a snapshot that is part of the script that finds the biggest value and places it on the top of the basket. In this snapshot one can see a general purpose variable ‘swap’ that is used as a temporarily variable for the sorting of the aforementioned values. With yellow color is the instruction that is executed at that time. The 5 values into the basket have been placed manually.

(b) some objects are given to this expert group, in order to construct programs for teaching their colleagues in the original Jigsaw groups. One such program is the following: there are 5 numbers and a red basket, implemented as a LIFO queue. That means, if we drag a number object into the red basket, it will be stored at the top of the corresponding queue. Moreover, if we click on the basket, then the script will look for the greatest number and place it on the top position of the basket (an indicative snapshot is illustrated in Figure 3c).

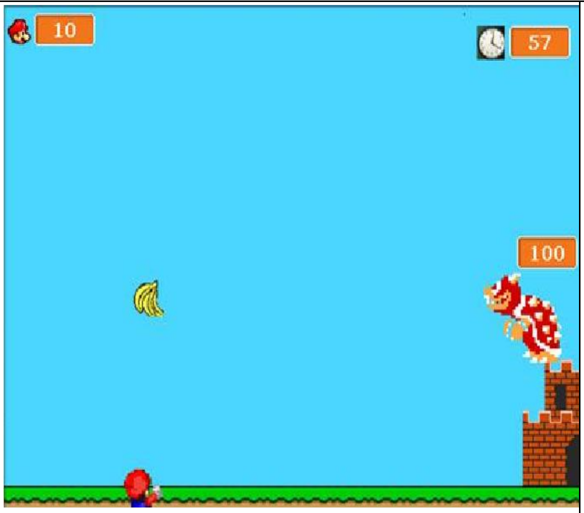
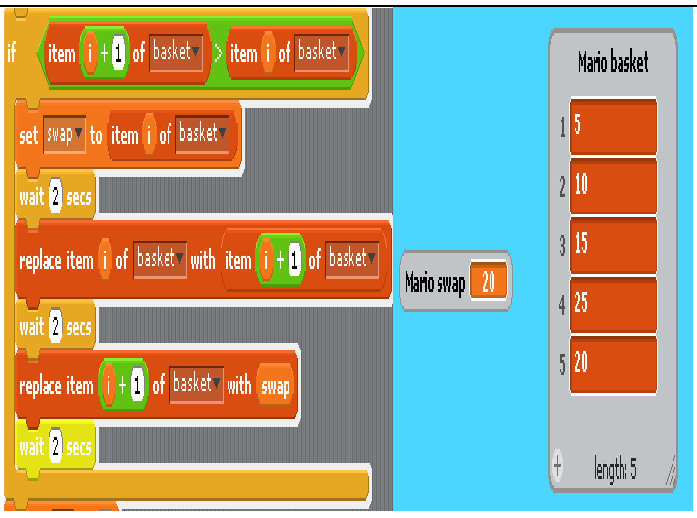


Fig. 3a. A snapshot of level 3



```
if < item (i + 1) of basket > item i of basket >
    set swap to item i of basket
    wait 2 secs
    replace item i of basket with item (i + 1) of basket
    wait 2 secs
    replace item (i + 1) of basket with swap
    wait 2 secs
```

Mario swap 20

Mario basket

1	5
2	10
3	15
4	25
5	20

length: 5

***Fig. 3b.** A snapshot indicating that position of value 25 repla 25 with value 20 , which is saved into the variable swap*

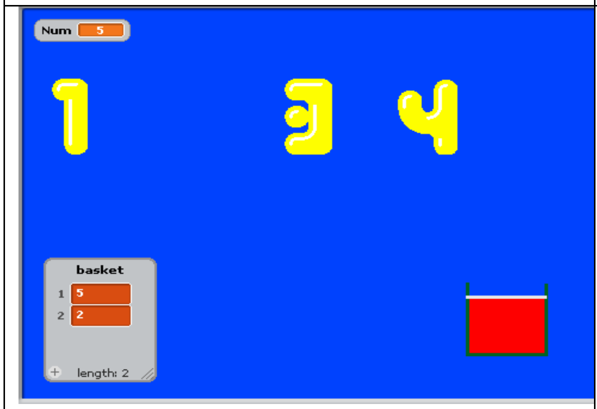


***Fig. 3c** A snapshot of level 3, about the add function of a LIFO queue*

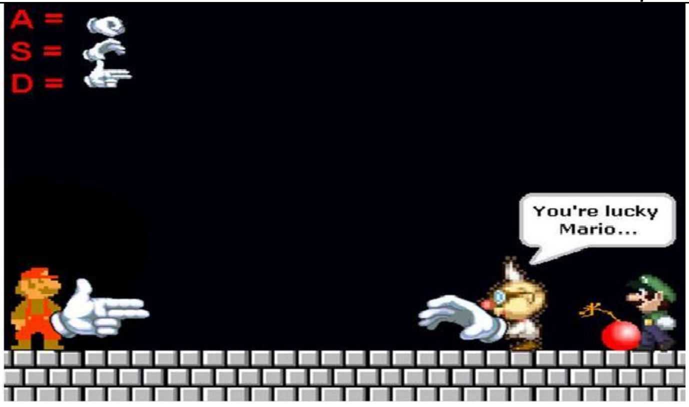


***Fig. 3d.** Snapshot of level 4*

D. Expert group – level 4

Here, Mario has to beat his opponent, Mojo, in the «Rock-Paper-Scissors» game. This game is played as follows: Players must count 5 seconds and then, simultaneously, they form, using their hands, one shape from the follows: rock, paper and a pair of scissors. Rock breaks scissors, rock is wrapped by paper and scissors cuts paper. When Mario loses, a bomb explodes and Luitzi (a friend of Mario) is killed. Otherwise, Luitzi is free. Mario’s choices are handled by the keys A, S, and D, as it is shown in Figure 3d, while Mojo’s choices are randomly performed by the game. Here, variables are used as signals for synchronization and notification. Both dialogue and appearance of their choices are implemented using these signals. To send and receive notification signals, Scratch provides 2 routines: «Broadcast» for sending a signal and «when I receive» for receiving a signal.

In order to understand this type of variables, a series of activities are suggested to be performed, like the following: this level starts with the variables-signals tangled, so the dialogue between Mario and Mojo is unsynchronized and the choice that the user makes for Mario (rock, paper or scissors) doesn’t correspond to the object that appears on the screen. Then, students are asked to perform the appropriate corrections.

Finally, students in this group are asked to construct the following program for teaching the original Jigsaw groups the type of variables mentioned in this level: this program consists of 2 persons, who participate in a dialogue and send signals to each other. The signals activate some additional objects to appear on screen and change their look, e.g. their sun glasses.

E. Return to the original Jigsaw groups

Students return back to their original Jigsaw groups to teach their classmates the implemented types of variables in programming, through, not only the initial Mario game, but also within the programs they constructed themselves. Each student will be assessed individually, by implementing a series of programming activities on each level. These activities involve the implementation of all types of variables that were mentioned within the Mario game. Indicative examples to assess students’ knowledge to the variables correspondent to: (a) level 1: students should be asked to change the motion of the object ‘mushroom’, in order to do free fall (using variables for the description of equations of motion). (b) level 2: students would be asked to add lives to Mario, when this level begins, and to lose a life each time the flying objects or the sea touch him (using general purpose variable – decrementer). (c) level 3: students should be asked to add some more Mario ‘scripts’ to describe, for example, his costumes when Mario loses, when Mario wins and when Mario attacks the dragon. This could be done by creating some new objects, representing Mario’s costumes in each of the occasions above, and by using signals in order to notify these new objects to appear on the screen, (d) level 4: students would be asked to modify some Mario scripts, so that Mario will not send signals to the ‘Rock’, ‘Paper’, ‘Scissors’ objects, but these will be incorporated into Mario’s new costumes. This task requires an understanding of the concept of structural variables (e.g. Mario's 'costume') by the students. Finally, students will be asked to collaborate in order to unify all levels into a single game by taking into account that each level follow the other and the lives of Mario aren't renewed at each level, but are inherited, giving continuity and meaning to the game.

4. SUMMARY AND PLANS FOR FURTHER RESEARCH

A set of cooperative learning activities are presented in this paper aiming to help high school students understand the concept of variable in programming. These activities are conducted through a game, constructed within the Scratch programming environment, and are based on a well-known and beloved hero of students, Super Mario. This game is structured in 4 levels, and students are divided into 4 groups, where each group focuses on a specific level that is dedicated for the learning of a specific class of variables. The activities are interactive, constructive, collaborative, and structured so as to excite the curiosity of students to experience and understand different aspects of variables in programming. To this direction, Scratch environment could be particularly helpful, because the programming procedure is done by constructing blocks of simple commands and not by writing text commands. Furthermore, Scratch is highly interactive with the users, providing them with opportunities to easily examine the functionality of specific command-blocks, by executing them independently and observing simultaneously the results on the screen next to the programming area. Moreover, the use of the Jigsaw collaboration method should be encouraging both; the active involvement of students in a collective effort, and the supportive role of the teacher, in order to help each group to produce the best results. Finally, more research is needed with real students, in order to evaluate the game and the activities mentioned above.

REFERENCES

1. **E. Aronson.** *History of the Jigsaw Classroom*, (1971). Retrieved at 10/06/2010 from: http://www.jigsaw.org/history.htm
2. **C. Crawford.** *The Art of Computer Game Design*, (1982). Retrieved at 10/06/2010 from: www.vancouver.wsu.edu/ fac/peabody/game-book/Coverpage.html
3. **A. McFarlane and S. Sakellariou.** The role of ICT in science education, *Cambridge Journal of Education*, 32(2), pp. 219-232 (2002).

4. **J. Ainley.** Playing games and learning mathematics. In L. P. Steffe & T. Wood (Eds.), *Transforming children's mathematics education: International perspectives*. Hillsdale, NJ: Erlbaum, pp.84-91 (1990).
5. **C. Anderson & C. Kamii.** Multiplication games: How we made and used them. *Teaching Children Mathematics*, 10(3), pp. 135-141 (2003).
6. **M. Csikszentmihalyi.** *Flow: The Psychology of Optimal Experience*. New York: Harper & Row, 1990.
7. **R. Rajaravivarma.** A games-based approach for teaching the introductory programming course. *Inroads ACM SIGCSE Bulletin*, Volume 37, Number 4, 2005 December, pp. 98 – 102 (2005).
8. **D. H. Jonassen.** Designing constructivist learning environments. *Instructional design theories and models*, 2, pp. 215-239 (1999).
9. **S.M. Land and M.J. Hannafin**. Student-Centered Learning Environments. In D.H. Jonassen and S.M. Land (Ed.), *Theoretical Foundations of Learning Environments*. NJ: Lawrence Erlbaum Associates, pp. 1-23, 2000.
10. **T. Bell, I. Witten & M. Fellows.** *Computer Science Unplugged*, (2002). Retrieved at 10/06/2010, from: http://www.unplugged. canterbury.az.nz
11. **Y.B. Kafai.** *The educational potential of electronic games: From games-to-teach to games-to-learn. Playing by the rules*, (2001). Cultural policy centre, university of Chicago. Retrieved June 10, 2010, from http://culturalpolicy.uchicago.edu/conf2001/papers/ kafai.html
12. **M.S. Fisch.** Making educational computer games educational. Proceedings of the *2005 conference on Interaction, design and children*, Boulder, Colorado, pp. 56 – 61(2005).
13. **J. Kirriemuir & C.A. McFarlane**. *REPORT 8: Literature Review in Games and Learning*, (2004). Retrieved June 10, 2010, from http://www.futurelab.org.uk/research/reviews/08_16.htm
14. **D. Oblinger.** The next generation of educational engagement. *Journal of Interactive Media in Education*, (8), pp. 1–18 (2004).
15. **M. Papastergiou & C. Solomonidou.** Gender and information and communications technology: Greek high school students' favourite internet activities. *Computers and Education*, 44(4), pp. 377–393 (2005).
16. **A. McFarlane, A. Sparrowhawk and Y. Heald.** *Report on the Educational Use of Games, TEEM* (Teachers Evaluating Educational Multimedia) (2002). Retrieved June 10, 2010, from www.teem.org.uk/
17. **M. Prensky**. *Digital game-based learning*, Mc Graw-Hill, New York, 2001.

18. **M. Papastergiou**. Digital game-based learning in high-school computer science education: Impact on educational effectiveness and student motivation. *Computers and Education*, 52(1), pp. 1-12 (2009).
19. **M. Kordaki.** A computer card game for the learning of basic aspects of binary system in primary education: design and pilot evaluation. *Education and Information Technologies* (Submitted for publication 2010).
20. **M. Scardamalia & C. Bereiter**. Computer support for knowledge-building communities. In T. Koschmann (ed.) *CSCL: Theory and practice of an emerging paradigm*, (pp.249–268). NJ: Erlbaum, 1996.
21. **A.G. Picciano.** Beyond student perception: Issues of interaction, presence and performance in an online course. *Journal of Asynchronous Learning Networks*, 6(1), pp. 21-40 (2002).
22. **P. Dillenbourg.** Introduction: What do you mean by collaborative learning?. In P. Dillenbourg (Ed.), *Collaborative learning: Cognitive and computational approaches*. Oxford: Pergamon pp. 1–19, 1999.
23. **D.W. Johnson & R.T. Johnson**. *Learning together and alone: Cooperative, competitive, and individualistic learning* (3rd ed.). Boston, MA: Allyn and Bacon, 1999.
24. **C. Haythornthwaite, M.M. Kazmer, J. Robins & S. Shoemaker**. Community development among distance learners: temporal and technological dimensions. Journal of Computer-Mediated Communication, 6 (1), (2000). Retrieved June 10, 2010: http://www.ascusc.org/ jcmc/vol6/issue1/ haythornthwaite.html.
25. **T. Koschmann.** *CSCL: Theory and practice of an emerging paradigm*. Mahwah, NJ: LEA, 1996.
26. **K. A. Brufee**. *Collaborative Learning: Higher Education Interdependence, the authority of knowledge*. Baltimore MD: The John Hopkins University, 1999.
27. **E. Lehtinen.** CSCL: an approach to powerful learning environments. In E. de Corte, L. Verschaffel, N. Entwistle, & J. van Merrieboer (Eds.), *Powerful learning environments: Unravelling basic components and dimensions*. Amsterdam: Pergamon, pp. 35-54 2003.
28. **P. McAndrew, P. Goodyear & J. Dalziel.** Patterns designs and activities: unifying descriptions of learning structures. *International Journal of Learning Technology*, 2(2-3), 216 – 242 (2006).
29. **V. Dagdilelis, M. Satratzemi & G. Evangelidis**. Introducing secondary education to algorithms and programming. *Education and Information Technologies*, 9(2), pp. 159–173 (2004).

30. **S. Hadjerrouit.** Towards a Blended Learning Model for Teaching and Learning Computer Programming: A Case Study. *Informatics in Education,* Vol. 7, No. 2, pp. 181–210 (2008).
31. **R. Brooks.** Towards a Theory of the Cognitive processes in Computer Programming. *Int. J. Human- Computer Studies*, 51, pp. 197-211 (1999).
32. **A. Pacheco, A. Gomes, J. Henriques, A-M. de Almeida & A-J. Mendes**. Mathematics and Programming: Some studies. *International Conference on Computer Systems and Technologies - CompSysTech'08*, V15-1, V15-6 (2008).
33. **E. Soloway & J.C. Spohrer.** *Studying the Novice Programmer*. NJ: Lawrence Erlbaum, Associates, 1988.
34. **A. Robins, J. Rountree, N. Rountree.** Learning and teaching programming: A review and discussion. *Journal of Computer Science Education*, 13(2), 449–450 (2003).
35. **P. Wegner, E. Roberts, R. Rada, A.B. Tucker.** Strategic directions in computer science education. *ACM Computing Survey*, 28(4), pp. 836–845(1996) .
36. **M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y.B-D. Kolikant, C. Laxer, L. Thomas, I. Utting & T. Wilusz.** A multinational, multi-institutional study of assessment of programming skills of firstyear cs students. *In ITiCSE-WGR* 2001, Canterbury, UK, pp. 125-180 (2001).
37. **M. Kordaki.** A drawing and multi-representational computer environment for beginners' learning of programming using C: Design and pilot formative evaluation. *Computers and Education*, 54(1), pp. 69-87 (2010).
38. **E. Lahtinen, K. Ala-Mutka & H. Järvinen**. A study of the difficulties of novice programmers. In ITiCSE '05: Proceedings of the *10th annual SIGCSE conference on Innovation and technology in computer science education*, Caparica, Portugal, ACM Press, pp. 14-18, 2005.
39. **J.L. Ford.** *Scratch Programming for Teens*. Canada: Course Technology PTR, 2008.
40. **T. Gallardo, L.A. Guerrero, C. Collazos, J.A. Pino & S. Ochoa.** Supporting JIGSAW-type Collaborative Learning. *System Sciences*, p. 8 (2003).
41. **M. Kordaki, H. Siempos & T. Daradoumis.** Collaborative learning design within open source e-learning systems: lessons learned from an empirical study. In G. Magoulas (Eds), *E-Infrastructures and Technologies for Lifelong Learning: Next Generation Environments*, IDEA-Group Publishing, 2010 (to appear).