Instr Sci (2017) 45:583–602
DOI 10.1007/s11251-017-9421-5

The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving

Sarantos Psycharis[1] · **Maria Kallia**[1]

Received: 8 April 2016 / Accepted: 22 July 2017 / Published online: 27 July 2017

Abstract In this paper we investigate whether computer programming has an impact on high school student's reasoning skills, problem solving and self-efficacy in Mathematics. The quasi-experimental design was adopted to implement the study. The sample of the research comprised 66 high school students separated into two groups, the experimental and the control group according to their educational orientation. The research findings indicate that there is a significant difference in the reasoning skills of students that participated in the "programming course" compared to students that did not. Moreover, the self-efficacy indicator of students that participated in the experimental group showed a significant difference from students in the control group. The results however, failed to support the hypothesis that computer programming significantly enhances student's problem solving skills.

Keywords Computer programming · Computational thinking · Problem solving · Reasoning skills · Self-efficacy · Mathematics

Introduction

Changes in organizational and technological structures in the workplace alongside increasingly new workplace demands and recent theories in teaching and learning, have put problem solving and reasoning skills as a top priority of curriculum development (Aukrust 2011; English and Halford 2012). One of the focal goals of school learning is the continuing advancement of a range of skills, abilities and attitudes, which will enhance one's ability to comprehend and control his or her cognitive processes (Leopold and Leutner

✉ Sarantos Psycharis
spsycharis@gmail.com

Maria Kallia
ma_kallia@yahoo.gr

[1] School of Pedagogical and Technological Education–ASPETE, 14121 Athens, Greece

2015). In the attempt to sustain a school environment contributing to learning, in an era that continues to change, new computational technologies and practices are implemented in teaching and learning (Psycharis 2016). Over the last years, the adequacy of ICT curriculum and the effects on children's skills have been questioned (Wells 2012). For this reason, many countries (e.g. UK, U S, Greece) have introduced a new initiative in their curriculum, which replaces the existing ICT courses with computer and computational science, in which students gradually learn how to program and be engaged in computational thinking (Jones 2011). Computational thinking-CT—(Wing 2006) is extensively used in education as it synthesizes critical thinking and existing knowledge and applies them to solve complex problems in mathematics, science and generally in STEM disciplines. CT has been coined by Wing and broadly speaking, it describes a set of thinking skills that are integral to solving complex problems using a computer. CT can be applied to many sciences (Engineering, Maths, Physics, Social sciences etc.).

Computational thinking is now recognized as a concept that encompasses the pervasiveness of computer science constructs and problem-solving strategies such as abstraction at different hierarchical levels, algorithmic thinking, automation, decomposition, modeling, patterns, recursion, scale, and symbolic representations (e.g. Cortina 2007; Guzdial 2008; Denning 2009; Grover & Pea 2013). The notion of computer programming in schools has been cultivated for many years with emphasis on the use of Logo as the programming language. In 1980, the innovative book of Seymour Papert *Mindstorms: Children, computers and powerful ideas* ushered in a new era, which involved children in the use of computer programming. Papert (1980) argues that children's involvement in programming could influence the way children learn and could have a positive impact on children's cognitive skills. It has been found that cognitive tools enable learners to work beyond their current limitations by providing scaffolding, so that learners can work on higher-order cognitive tasks (Lajoie 1993; Jonassen 2003).

The term 'problem-based learning' (PBL) refers to instructional approaches that use "real world," simulated, contextualized problems of practice to motivate, focus and initiate content learning and skill development. In PBL environments, students have opportunities to practice applying their content knowledge and workplace skills, while working on authentic, contextualized problems and projects (Spector 2003). Problem solving has a special importance in the study of mathematics and is related to programming. A primary goal of mathematics teaching and learning is to develop the ability to solve a wide variety of complex mathematic problems (Stanic and Kilpatrick 1988). According to Soloway (1993), during the programming, the learner uses powerful problem-solving/design/thinking strategies. This is because when learners program, they first need to find a solution to a problem from mathematics and then to reflect on how to communicate their solution to the computer, using syntax and grammar, through an exact way of thinking (Papert 1980; Szlávi and Zsakó 2006). Syslo and Kwiatkowska (2006) state that through programming students also acquire a sense of mastery over a technological instrument and establish contact with some of the deepest ideas of different disciplines such as mathematics. Although PBL helps students acquire the knowledge and skills required of professionals in the workplace, competent task performance also requires self-efficacy (Schunk 1989). Because of PBL's use of problems of practice to drive learning, students engaged in a PBL environment should experience performance accomplishments that have an overall impact on their self-efficacy.

The purpose of the present research is to investigate the impact of computer programming curriculum intervention on reasoning skills, problem solving skills in mathematics and self-efficacy in mathematics, of 16–17 year-old students who were taught

computer programming (Greek version of Pascal) during the last year of high school in Greece, following the Greek Curriculum. To achieve that, an experimental research design was used to investigate students' problem solving, reasoning skills and self-efficacy. Specifically, the research questions that this study seeks to answer are the following:

1. Does teaching mathematics in conjunction with computer programming affect students' reasoning skills in general, and if so, to what extent?
2. Does teaching mathematics in conjunction with computer programming affect students' self-efficacy in mathematics, and if so, to what extent?
3. Does teaching mathematics in conjunction with computer programming affect students' problem solving skills in mathematics, and if so, to what extent?

Theoretical background

Problem solving, reasoning skills and self-efficacy

Cognitive skills are skills that are connected *with thinking and knowing—the skills required for children to understand language and numbers, to reason and problem solve, and to learn and remember* (Subrahmanyam et al. 2000). Bruner (1957) defines reasoning skills as the process of making conjectures and conclusions from information and they are divided into two important categories, namely: deductive reasoning and inductive reasoning (Sternberg 2008; Lohman and Lakin 2009). Reasoning is also considered a critical aspect of analytical thinking and problem solving (Robbins 2011).

OECD (2013), in PISA 2012 assessment and analytical framework, defines problem solving as a competency which engages a person in the cognitive processes of comprehension and resolving of a problematic situation with a non obvious solution. According to Schoenfeld (1983), "a problem is only a problem, if you do not know how to go about solving it. A problem that has no surprises in store, and can be solved comfortably by routine or familiar procedures (no matter how difficult!) it is an exercise".

Problem solving skills now form an integral part of the curriculum and the need for students to become successful problem solvers has become a dominant matter in many educational themes (Foshay and Kirkley 2003). Problem solving skills should be deemed of great importance if the society needs students to function effectively and efficiently in a world of high demands (Barr and Stephenson 2011; Green and Gilhooly 2005). According to Green and Gilhooly (2005), the ability to efficiently and effectively interpret new information will become more important than the specific knowledge someone possesses. This is an important reason why many countries have incorporated problem solving as a central competency in their curriculum. It is argued that the possession of problem solving skills will help children acquire new knowledge and will provide them with the necessary skills to successfully enter the society, conduct personal activities, adapt to new conditions and confront life difficulties (Lesh and Zawojewski 2007).

Denning (2009) discussed the relation of CT with problem solving, and he considered CT as a mental orientation to formulate problems, as conversions of some inputs to outputs and looking for algorithms to perform the conversions. CT can help towards problem-solving through various techniques and strategies. Techniques and strategies may include organising data logically, breaking down problems into parts, defining abstract concepts and designing and using algorithms, patterns and models in order to solve problems. According to researchers, when computers are used in problem solving, the need for CT (in

its broader meaning, which includes abstract and logical reasoning, algorithmic thinking and not only computation techniques) is a prerequisite for problem solving (Halpern 2003; Matlin 2005). CT does not propose that problems need to be solved in the same way a computer tackles them, but rather it encourages the reasoning skills using computer science concepts, algorithms and programming techniques. In the early 1980s, computer programming was first recognized as an effective tool with a high cognitive value for teaching the basic concepts that can be applied to mathematics, physics and logic (Papert 1980; Howe et al. 1989) and for transferring problem-solving skills to other fields of knowledge (Ennis 1994).

Soloway (1993) argues that in learning to program one learns powerful problem-solving/design/thinking strategies. This is because when students program, they first need to find a solution to a problem, and then to reflect on how to communicate their solution to the machine, using syntax and grammar, through an exact way of thinking (Papert 1980; Szlávi and Zsakó 2006). Programming involves the ability to generate a solution to a problem and generating solutions means that one of the learning outcomes is the ability to solve problems (Saeli et al. 2011). According to Papert (1980), when students create a program they acquire a sense of mastery over a technological instrument and they establish contact with some of the deepest ideas of different disciplines such as science and mathematics.

Programming is considered a complex cognitive ability and according to Kagan (2006) it can be an excellent example of how someone can obtain such a compound cognitive skill. Learning how to program requires strategy, planning and logical thinking skills and, as such, it provides a productive field for developing and exercising problem solving skills, higher order thinking and metacognitive skills (Lavonen et al. 2003). Writing a computer program requires some degree of interpretation, abstraction, logical reasoning, comprehension of the structure of the program and altering the source code in order to be functional. It also requires skills of trouble shooting, discovering errors in the code, modifying the initial idea to match a specific situation, and generally, practical competencies that cannot be developed through theory (Govender 2007). Thus, the process of programming is believed to favour skills like creativity, logical reasoning, skills for design and innovative thinking and communication (Falkner and Palmer 2009). There are many similarities in the above process with models of cognitive functions, which make computer programming an activity which enhances cognitive performance. When someone creates computer programs that represent the complexity of human thinking then she/he understands this behaviour and learns about her/his own thinking processes (Pea et al. 1985). Furthermore, the processes of decomposing compound problems into sub-problems, generating complete and accurate solutions for the sub-problems and testing and re-testing the problem solution for the correctness and efficiency, demand high problem solving skills and make computer programming a vehicle for exercising analytical and critical thinking for a variety of problems (Michalewicz and Michalewicz 2008). One of the core aims of using programming is also to engage students to think computationally and acquire skills to develop solid solutions through understanding of concrete problems using the analytical and critical thinking. Recent studies in this field address the necessity for students to be trained in thinking computationally before learning to program (Denning 2009).

Many researchers have tried to explore student engagement in programming for the purposes of mathematical education. DiSessa (2000) argues that programming turns analysis into experience and allows a connection between analytic forms and their experiential implications that mathematics can not touch. Along the same lines, Sendov and Sendova (1995) had stressed the affordance of a programming language for expressing, elaborating and communicating ideas. Some studies that place programming at the heart of their

endeavour are taking a further step away from recasting mathematical activity in new representational forms, and towards recasting the nature of school mathematical activity itself.

The key areas of the educational studies on computer programming have focused mainly on the possible effects that computer programming has on students' skills and whether the skills learned during programming instruction could be transferred to other non-programming problems (Jonassen and Reeves 2001). The results of these studies are controversial. In specific, there are studies supporting that computer programming can impact students' skills, whereas others fail to support the same argument. Some findings of the studies supporting that programming affects students' skills are the following:

1. Programming improves divergent thinking and metacognitive ability (Clements and Gullo 1984).
2. Programming encourages problem solving ability (Linn 1985; Miller et al. 1988; Clement et al. 1990; Tu and Johnson 1990; Taylor et al. 2010; Pardamean et al. 2011).
3. Programming improves reasoning skills (Tu and Johnson 1990; Fox and Farmer 2011).
4. Programming enhances collaboration and mathematical thinking (Taylor et al. 2010).
5. Programming enhances creativity (Pardamean et al. 2011).

Among the studies failing to support that computer programming can impact students' cognitive skills are the studies of Pea and Kurland (1984), Pea et al. (1985), VanLengen and Maddux (1990) and White (2011). These researchers argue that there is a need for more empirical data to address the subject, that children did not generalize specific program concepts and procedures, the short time of the treatment in the experimental group, no teacher—student interaction, and non use of problem solving strategies.

Moreover, there is a lot of research on problem solving and self-efficacy in Mathematics. Bandura (1994) defines self-efficacy as "people's beliefs about their capabilities to produce designated levels of performance that exercise influence over events that affects their lives" (Bandura 1994). Self-efficacy plays an important role in Education since students who believe in their capabilities do not avoid difficult tasks, but rather they consider them as challenges that need to be addressed (Bandura 1994). This means that self-efficacy influences the amount of effort students will demonstrate while solving a problem and as such, it can motivate students to learn better and be aware of the abilities, strength and weaknesses (Gandhi and Varma 2009).

In search of practices and strategies to influence students' Mathematics achievement, many researchers have studied the relationship between self-efficacy and Mathematics achievement and they have shown that self-efficacy influences mathematical achievement (Bandura et al. 1996; Pajares and Urdan 2005). Therefore, the role of self-efficacy is significant for enhancing learning and engaging students more effectively on Mathematics problems (Caswell and Nisbet 2005). According to research, under PBL, students become more intrinsically motivated, showcase more independent learning, report higher levels of self-efficacy, have better-developed meta-cognitive skills, and are more autonomous than students who are not instructed under PBL. The development of these strengths can promote higher levels of achievement (Ali et al. 2011; Schmidt et al. 2011).

Methodology

The focal purpose of this study is to investigate the effects of computer programming curriculum on students' reasoning skills, problem solving skills and self-efficacy in Mathematics through the co-teaching of these two school subjects. The problem solving

skills focused on Mathematics problem solving and were assessed with a test. This test consisted of one problem adapted from the Greek national examination board and assessed students' performance in a specific taught unit in Mathematics. The reasoning skills were assessed with the Cornell Reasoning Test, while the self-efficacy was tested with the Motivated Strategies for Learning Questionnaire (MSLQ), developed by Pintrich, Smith, Garcia and McKeachie in 1991.

Prior to administering the tests to students, both the Cornell Reasoning test and the MSQL were translated into Greek by an English Language teacher and were piloted. The aim of the pilot study was to try out the research approach and to identify potential problems that may affect the quality and validity of the results (Blessing and Chakrabarti 2009). The results of the pilot test revealed some problems with the translation from English to Greek that needed clarification. These problems were taken into consideration and the problematic areas were amended in order to be better understood by the students.

After obtaining the necessary permissions, the students who participated in the study were provided with informative consent forms, which were completed by their parents.

Sample

The participants were 66 students in the final class of a state high school (Lyceum). For the research study, the students were separated into two groups (33 students per group)–the experimental group and the control group. The method for the sampling was the convenient sampling. In this type of sampling, a portion of the target population is selected based on certain practical criteria, such as geographical proximity, availability at a certain time, easy accessibility, or the willingness to volunteer (Farrokhi 2012). The experimental group comprised students that had selected the Informatics orientation of Lyceum, while the control group comprised students of the technology orientation. The main difference between these two orientations is that students in the Informatics orientation are taught the course 'Development of Applications in Computer-Programming Environments' while students in the Technology orientation are taught 'Chemistry' and 'Electrology'. Both orientations have as common courses 'Mathematics' and 'Physics' with the same syllabus.

Among the courses that exercise reasoning and problem solving skills are the Mathematics and Physics courses. Acknowledging that both groups are taught these courses by the same teacher and are taught the same curriculum, any difference in students' reasoning and problem solving skills should be accounted to the teaching and learning process of other courses. In the experimental group, the course 'Development of Applications in Computer-Programming Environments' we consider it has a strong focus on the development and exercise of the analytical way of thinking and on practising techniques to develop logical reasoning and algorithmic procedures and any difference in the reasoning or problem solving skills between the two groups could be associated with the course of computer programming.

Thirty-three (33) students from the Informatics orientation formed the experimental group and thirty-three students from the Technology orientation formed the control group. In the experimental group, the number of girls was 13 while the number of boys was 20. The number of girls that formed the control group was 16 and the number of boys was 17. In the experimental group, eight students were at the age of 16 and 25 students were at the age of 17. In the control group, seven students were at the age of 16 and 26 students were at the age of 17.

Design and procedure

The main purpose of this research study was to examine the relationship between computer programming and reasoning skills, problem solving and self-efficacy in Mathematics of last year high school student. To investigate this relationship, the type of the research design adopted was the quasi-experimental design and in particular, the non-equivalent control group pre-test/post-test design. In non-equivalent group designs, different groups receive different treatments and the effectiveness of a treatment is evaluated by comparing the performances of the two groups (Millsap and Maydeu-Olivares 2009). This design requires a pre-test, in order to have an indication of how similar the control and the experimental group were before the intervention, as well as a post-test (Fife-Schaw 2000). With this design, both the control group and the experimental group are compared; however, the groups are chosen and assigned out of convenience rather than through randomization. Applying this design in this study, two pre-existing groups were required, the experimental group which consisted of students in the Informatics Orientation and the control group which consisted of students in the Technology Orientation. Moreover, both groups were pre-tested in order to be sure that they did not differ significantly before the intervention took place.

No particular intervention followed for the testing of reasoning skills of students. That was because we were interested in investigating whether computer programming courses, as already delivered by the current curriculum, could have any impact on students' reasoning skills. The students' reasoning skills were measured before and after a 90 mins computer programming class instruction. For the Mathematics problem solving skills and self-efficacy, an intervention course was designed for the experimental group. Both groups were taught the same part of the Mathematics syllabus, which had been selected as appropriate for implementation in a computer programming environment.

Specifically, in the experimental group, the Math teacher collaborated with the computer programming teacher to teach the specific part of the syllabus. In detail, the Math teacher explained the problem under study and then an exercise was assigned to the students. Students developed a computer program in the form of a source code based in algorithmic thinking which was used to solve the Mathematic problem set by the Math teacher, and generalised their solution so as to handle similar problems. Then the code was transferred into the Matlab software tool, where an appropriate graphical representation of the solution was created to illustrate the correctness of the code and the solution of the exercise. In the end, the students together with the Math teacher discussed and reflected upon the solution and the problem solving strategy adapted to solve the exercise selected. They also discussed and reflected with the Computer science and Maths teacher upon the solution and the methodology and analytical thinking adapted to create the computer program. This intervention took place in a three-hour course instruction. The same exercise was distributed to the control group by the same Math teacher but this time students followed a standard operational procedure, which is rather a methodological approach to solve problems. Having solved the problem, the Math teacher distributed other similar exercises which the students solved following the necessary steps. The course lasted approximately one and a half hour. Before and after the completion of the aforementioned process, both the experimental group and the control group were given a test adapted from the national examination board for K-12 students based on the specific syllabus.

Students' scores were compared before and after the programming course. In particular, the pre-test scores of students' reasoning, problem solving skills and self-efficacy were

compared with the corresponding post-test scores. A paired t test could have been employed to compare the means of the same group in the pre-test and post-test and the independent measures t-test could have been used to assess the significance of the difference between the means of the two groups. Since the data were not normally distributed, the Wilcoxon signed ranked test and the Mann–Whitney U test were used instead. The Wilcoxon signed ranked test is a non-parametric test and was used to test the difference between the median calculated more than once for the same group while the Mann–Whitney U test was used to test if the data collected from two independent groups differ significantly.

Measurements

Reasoning skills

The deductive Reasoning test that was employed in this study was the Cornell Conditional Reasoning test Form X, published by the Illinois Critical Thinking Project, Department of Educational Policy Studies (McLellan 2009). The test consists of 72 multiple choice items covering the 12 principles defined by Ennis and Paulus (1965). Every question is described with one or more sentences and an extra sentence is provided for which students must decide its validity according to what they have read. The Cornell Reasoning Test was administered to students in October as a pre-test and in April as a post-test (after 90 teaching hours) both in the experimental group and the control group. At the beginning of the assessment, students completed an initial questionnaire which included background questions determining the characteristics of the students. Students were asked to provide information regarding their age and their familiarity with any programming language. When the first part was completed, the Cornel Reasoning Test was distributed to students for completion.

The main reason why the Cornell Critical Thinking test (CCTT) was employed was that it is one of the most commonly used instruments to assess reasoning skills (Ennis et al. 1985). Moreover, the specific instrument is used to measure the reasoning skills of high school students which comprised the sample of our study. The only drawback of this test is that it entails too many questions (72), and so students found it tedious and time consuming.

Problem solving skills and self-efficacy

A Motivated Strategies for Learning Questionnaire (MSLQ) and a Mathematics Assessment Test for the learning performance was administered to students in the experimental and the control group before and after a specific structured course in computer programming, so that students included in the experimental group could write source code. The aim of this course was to engage students in concepts of Statistics using a computer programming environment. The reason why this specific course was selected to be implemented in a computer programming environment was that it was considered the most suitable for demonstrating a Mathematical-statistical problem that could be solved in a computer programming environment and furthermore to transfer the program code to Matlab tool and depict its solution graphically. The control group was given the same problem and students were asked to solve this with the traditional way.

The questionnaire that was employed for the study of self-efficacy was the Motivated Strategies for Learning Questionnaire (MSLQ), developed by Pintrich et al. (1991), which

was designed to measure students' motivation, self-efficacy and self-regulation learning into a specific course. The MSLQ consists of 81 items which are separated into two sections, the learning strategies and motivation sections, which are further divided into the sub-section of students' goals and value beliefs for a course, their self-efficacy in this course and their anxiety about tests in this course. The sub-section which was used for this study was the self-efficacy which is part of the MSLQ. Scores are calculated by the mean of the items that are included in the scale (eight items for self-efficacy) (Artino 2006). Before the beginning of the course, MSLQ was administered to students to measure their self-efficacy in Statistics. A three-hour course was carried out on a weekly basis, for 90 h, and students of the experimental group had the task to apply the concepts of the programming language in order to solve problems and exercises in Statistics.

The reasons why the Motivated Strategies for Learning Questionnaire was employed is that it appears to be a very sound, flexible instrument, measuring what it is supposed to measure (Artino 2006). Moreover, many other researchers have used this instrument for their studies, which testifies its validity and reliability (Doubé and Lang 2012). On the contrary, the most important weakness of the questionnaire is the low internal reliability of some subscales (Artino 2006). However, the subscale that was used in this study is not included into the ones of low internal reliability. Finally, the reason that tests were chosen as the research tool is because they are suitable for demonstrating students' acquisition of knowledge or their ability to process and use knowledge and they allow the comparison between different or the same group over time (Woosley 2012).

After the intervention, both the experimental group and the control group were administered the MSQL and the mathematics Assessment test. The mathematic test that was employed in this study was selected from the test collection of the Greek national examination board. It consists of one statistical problem for which students should provide their solution. Problem was complex and was presented in structural form. The Mathematics teacher who proposed the tested exercises has more than 35 years of teaching experience in this specific course and his advice and contribution to what should be tested was valuable for the needs of this experiment.

Results

In this section, we turn to our research questions by examining students' scores on the reasoning skills, problem solving skills and self-efficacy at pre-test and post-test.

Self-efficacy

To statistically analyse students' responses on the MSLQ questionnaire, the SPSS software was used. Firstly, the students' pre-test scores and the students' post-test scores were recorded and both scores served as inputs in the SPSS program. To determine the relationship between computer programming instruction in parallel with Mathematics syllabus and students' self-efficacy in Mathematics, the Wilcoxon signed rank test was used, since the responses of the pre-test and post-test came from the same group of students and the data were not normally distributed. The Mann–Whitney U test was used to compare the scores between the two groups since the data were not normally distributed. To ensure that the two groups did not differ significantly before the beginning of the intervention, we compared the two groups' scores in the pre-test. The Mann–Whitney U test was applied to

ensure that the two groups did not differ significantly in the pre-test. As indicated from the p value (0.817), the two groups did not significantly differ in the pre-test scores.

Wilcoxon sign rank test

To examine whether the difference between the pre-test and the post-test in the same group was significant, we used the Wilcoxon signed rank test. The null hypothesis that was tested is: "There is no significant difference between the pre-test score and the post-test score for the control group". Since the p value (0.173) is greater than the α value (0.05), the null hypothesis cannot be rejected. Thus, the difference between the mean score in the pre-test and in the post-test for the control group is not significant. For the experimental group, the null hypothesis that was tested is: "There is no significant difference between the pre-test score and the post-test score for the experimental group". The p value equals zero (0.000) and thus the null hypothesis is rejected which means that the difference between the pre-test score and the post-test score is significant. Specifically, the Wilcoxon signed-rank test showed that the instruction of Mathematics in parallel with computer programming did elicit a statistically significant change in the self-efficacy of the students ($Z = -4.293$, $p = 0.000$).

Mann–Whitney U test

To test whether the two groups differ significantly in the post-test, the independent t test could not be employed since the data of the groups were not normally distributed. For this reason, the non-parametric Mann–Whitney U test was used. Specifically, the null hypothesis that was tested is: "There is no significant difference between the scores of the experimental group and the control group in the post-test". Considering the p value (0.047), the null hypothesis is rejected. Specifically, the Mann–Whitney U test showed that the instruction of Mathematics in parallel with computer programming course did elicit a statistically significant difference in the self-efficacy of the students in the experimental group from students in the control group ($Z = -1.987$, $p = 0.047$).

We also calculated the size effect (r) using the formula

$$r = \frac{Z}{\sqrt{N}},$$

where, for the Mann–Whitney test $N = N1 + N2$, $N1 =$ control group size, $N2 =$ experimental group size, and for the Wilcoxon test, $N = 2 \times$ (size of the group)

1. The Wilcoxon signed-rank test showed that the instruction of Mathematics in parallel with computer programming did elicit a statistically significant change in the self-efficacy of the students in the experimental group ($Z = -4.293$, $p = 0.000$, $r = 0.528432$). The value of $r = 4.293/\sqrt{(2 \times 33)} = 0.528432$.
2. The Wilcoxon signed ranks test showed that there was not a statistically significant change in the performance of students in the control group ($Z = -1.362$, $p = 0.173$, $r = 0.167651$).
3. The Mann–Whitney U test showed that the instruction of Mathematics in parallel with computer programming course did elicit a statistically significant difference in the self-efficacy of the students in the experimental group from students in the control group ($Z = -1.987$, $p = 0.047$, $r = 0.243475$). The value of $r = 1.987/\sqrt{(33 + 33)} = 0.243475$.

Problem solving-mathematics test

In this section, the results of the Mathematics test are provided, which measure the learning performance of students. Marking was based on the different subcomponents of the problem after the engagement of students in programming techniques, use of algorithms and development of source code for the experimental group. Both the experimental group and the control group completed the test.

To ensure that the two groups did not differ significantly before the beginning of the intervention, we compared the two groups' scores in the pre-test with The Mann–Whitney U test. The test revealed that the two groups did not differ significantly in the pre-test (p value = 0.370).

Wilcoxon signed ranks test

To test the difference in the score of the pre-test and the post-test of the same group, the Wilcoxon Signed Ranks test was used. The test revealed that the difference in the pre-test and post-test scores of the control group is not significant since the p value (0.065) is greater than the α value (0.05), whereas the experimental group score showed a significant difference as the p value (0.020) is less than the α value (0.05). Specifically, the Wilcoxon signed ranks test showed that the instruction of Mathematics in parallel with computer programming course did elicit a statistically significant change in the performance of students in the experimental group ($Z = -2.324$, $p = 0.020$).

Mann–Whitney U test

To test the significance between the experimental group score and the control group score in the post-test, the Mann–Whitney U test was employed. The null Hypothesis states that "There is no difference between the control and the experimental group in the post-test". Since the p value is equal to 0.357 and is greater than the α value (0.05), the null hypothesis is not rejected.

We present the results for the size effect (r).

1. The Wilcoxon signed ranks test showed that the instruction of Mathematics in parallel with computer programming course, did elicit a statistically significant change in the performance of students in the experimental group ($Z = -2.324$, $p = 0.020$, $r = 0.286065$).
2. The Wilcoxon signed ranks test showed that there was not a statistically significant change in the performance of students in the control group ($Z = -1.848$, $p = 0.065$, $r = 0.227473$).
3. The Mann–Whitney U test showed that there was not a statistically significant difference in the problem solving skills of the students in the experimental group from students in the control group ($Z = 0.921$, $p = 0.357$, $r = 0.113367$).

Reasoning test

In this section, the results of the Reasoning test are provided. To ensure that the two groups did not differ significantly before the beginning of the computer programming course we compared the two groups' scores in the pre-test. The Mann–Whitney U test was executed to ensure that the two groups did not differ significantly in the pre-test. The test revealed

that the two groups did not differ significantly before the beginning of the programming courses (p value = 0.335).

Wilcoxon signed ranks test

To test the difference between the means of the control group in the pre-test and the post-test, the Wilcoxon signed ranks test was used. The null hypothesis that was tested is: *There is no significant difference in the pre-test scores and in the post-test scores of the control group*. The test showed that the null hypothesis is not rejected since the p value is 0.166.

To test the difference between the mean score of the experimental group in the pre-test and the post-test, the Wilcoxon signed ranks test was used. The null hypothesis that was tested is: "There is no significant difference in the pre-test scores and in the post-test scores of the experimental group". The test revealed that the null hypothesis is rejected since the p value equals 0.000.

Mann–Whitney U test

For the significance between the experimental group score and the control group score in the post-test, the Mann–Whitney U test was employed. The null Hypothesis that was tested is: "There is no significant difference between the control and the experimental group in the post-test". Since the p value is equal to 0.048 the null hypothesis is rejected. Specifically, the Mann–Whitney U test showed that the instruction of computer programming did elicit a statistically significant difference in the reasoning skills of the students in the experimental group from students in the control group ($Z = -1.978$, $p = 0.048$).

We present the results for the size effect(r)

1. The Wilcoxon signed ranks test showed that the instruction of computer programming did elicit a statistically significant change in the reasoning skills of students in the experimental group ($Z = -3.943$, $p = 0.000$, $r = 0.48535$).
2. The Wilcoxon signed ranks test showed that there was not a statistically significant change in the performance of students in the control group ($Z = -1.387$, $p = 0.166$, $r = 0.170728$).
3. The Mann–Whitney U test showed that the instruction of computer programming did elicit a statistically significant difference in the reasoning skills of the students in the experimental group from students in the control group ($Z = -1.978$, $p = 0.048$, $r = 0.243475$).

Discussion and conclusions

Self-efficacy results

The first research question of this study focused on investigating whether teaching computer programming in conjunction with Mathematics has a significant difference on students' self-efficacy in Mathematics. To answer this question, the Motivated Strategies for Learning Questionnaire (MSLQ) developed by Pintrich et al. (1991) designed to measure students' motivation, self-efficacy and self-regulation learning to a specific course was employed. The sub-section used for this study was the self-efficacy part which consists of

eight questions, while the answers are presented as a 7-point Likert scale. The aforementioned results were collected in order to answer the research question:

"Does teaching mathematics in conjunction with computer programming affect students' self-efficacy in Mathematics, and if so, to what extent?" To address this question, the Mann–Whitney U test was used to compare students' score for the two groups and the Wilcoxon signed ranks test was employed to test for the difference between the same groups.

The scores obtained by each group in the pre-test and the post-test were compared within the same group, in order to be able to draw conclusions about whether and how the computer programming course affected or not the experimental group. The data indicated that there was not a significant difference in the scores of the control group, in the pre-test and in the post-test. On the contrary, the analysis of the results for the experimental group specifies that a computer programming combined with mathematics syllabus resulted in a significant difference on students' self-efficacy in mathematics and thus there is an indication that computer programming may affect students' self-efficacy.

Moreover, in order to be certain that the difference between the two groups is significant, a comparison in the post-test scores of the two groups was implemented using the Mann–Whitney-test. As indicated by the results, there is a significant difference between the two group scores. From the aforementioned results, the answer to the research question number one is affirmative, which signifies that teaching computer programming in conjunction with Mathematics affects students' self-efficacy in Mathematics.

These findings can be interrelated to the investigation carried out by Güzeller and Akin (2012) in which they concluded that teaching mathematics with ICT tools improves students' self-efficacy in Mathematics. The previous result was also supported by the study of Bescherer and Zimmermann (2013) that evaluated university students' mathematical self-efficacy when they use ICT to support learning in mathematics. Their study indicates that students' mathematical self-efficacy was significantly higher when employing ICT tools than without. This study further contributes to the literature by adding that computer programming may also affect students' self-efficacy in Mathematics.

Self-efficacy has been shown to play an important role on students' mathematical achievements (Bandura et al. 1996). Its importance is also stated by Pajares and Urdan (2005) who found that 25% of students' success depends on their self-efficacy. Therefore, it is important that teachers and students employ tools that will improve students' self-efficacy, especially in a subject such as Mathematics, which is regarded as a difficult subject to comprehend.

Problem solving skills results

The second research question of this study is focused on finding out whether teaching computer programming in conjunction with Mathematics has a significant difference on students' problem solving skills in Mathematics for the experimental group and the control group. As the results indicate, the difference between the two scores of the control group is not significant. On the other hand, the results for the experimental group indicate that the difference between the scores of the experimental group is significant. To test the difference of the post-test scores of the two groups, the Mann–Whitney U test was used. This test revealed that the null hypothesis could not be rejected, which indicates that the instruction of mathematics in parallel with computer programming course did not yield a statistically significant difference in the performance of students in the experimental and the control group. The aforementioned data were used to answer the research question: "Does

teaching Mathematics in conjunction with computer programming affect students' problem solving skills in Mathematics, and if so, to what extent?". On the one hand, the failure to reject the null hypothesis designates that there was not a significant difference between the two groups and thus the computing course in relation with the mathematics syllabus did not help students in the experimental group achieve high enough marks to highlight the difference with the control group. On the other hand though, the results revealed that the experimental group achieved a higher mean score than the control group. This higher mean score, however, is not significant enough to confirm the hypothesis that computer programming improves students' problem solving in mathematics, but it is persuasive to suggest that it might, as the difference in the experimental group score in the pre-test and in the post-test is significant. Although this may be true, further experiments should be conducted. According to Caprara et al. (2011), students' self-efficacy can contribute to their academic achievements. Since in this study, the findings reveal high self-efficacy in students who are being taught mathematics with computer programming, similar results can be anticipated in further research for the problem solving ability. Moreover, e research findings for the experimental group mirror the results of Clement et al. (1990) who found that students can better understand and solve a mathematic equation in the context of computer programming. Likewise, the study of Taylor et al. (2010) investigates how engaging a computer programming environment is for students and signifies that computer programming environments enhance collaboration between students and advance their mathematical thinking and problem solving (Taylor et al. 2010). Moreover, this study's conclusions reflect the findings of Pardamean et al. (2011) who investigated the effects of Logo programming on children's problem solving skills and creativity. Their findings indicate that students who have been instructed programming language courses (experimental group) had achieved significantly higher scores than students that did not (control group) in both creativity and problem solving skills. Finally, Maloney et al. (2004) hypothesised that if learners work on individually meaningful computing projects, they will develop technological fluency, mathematical and problem solving skills.

Using programming language to solve a problem, does not imply that the learner uses only algorithmic process. In developing a source code to solve a problem, students use logical and abstract reasoning, decomposition and algorithmic thinking if necessary, i.e. they are engaged in CT as a new way of thinking. It should be constantly highlighted that engagement in using programming does not mean that students just write source code, but it involves aspects of computational thinking like abstract and logical thinking and demands skills in order to develop solid solutions through understanding of concrete problems in Maths, Science and generally in STEM disciplines (Denning 2009).

According to Kazimoglu et al. (2011), CT can enhance a problem based approach and there is no better way to learn it explicitly than through programming. Programming also employs main aspects of CT and the knowledge gained through the experience of tackling programming challenges—both explicit and tacit—can provide a framework not only for computer science, but for any field from Maths, Science, Engineering disciplines. The art of programming requires creativity and inventiveness, logic, algorithmic and abstract thinking and an appreciation of the recursive nature of this process, as the student learns from his/her failures, refines his/her work and gets a deeper understanding of the problem to solve in STEM disciplines (Einhorn 2012).

While there are research studies which indicate the positive impact of computer programming on problem solving skills, there are also studies that indicate that computer programming is ineffective in gaining problem solving skills. There is a lot of discussion around the reasons why computer programming is ineffective for the acquisition of

problem solving skills in general and in mathematics. Some research studies indicate that while computer programming alone is ineffective in teaching problem skills, when computer programming is paired with systematic problem solving instruction, learners demonstrated significant learning gains (Dalton and Goodrum 1991; Unuakhalu 2008).

We consider that one of the possible reasons why in our research study students' engagement in programming was not effective in problem solving in mathematics, was that students were not trained in the problem solving process in alignment with the CT issues; namely how CT could impact the solving process which includes analysis and organization of data, data abstractions, formulation of the problem, algorithmic thinking, decomposition, identification of possible solutions and generalization to other problems. Most students faced the problem rather as an exercise rather than as a problem. Involvement in CT could help students to conceptualize the different steps in problem solving and not face the problem simply as an exercise, but in this case a further examination is necessary because the difference could be attributed to the training in CT and not in the engagement in the development of programming code. Another possible reason for the failure of the impact of programming in problem solving, already stated in research (Kordaki 2012), could be the lack of instruction strategies related to computer programming and mathematical problem solving. Research suggested a set of categories of learning activities that can be delivered to learners via programming. Researchers categorized these learning activities under 11 headings, including creative activities, problem solving activities and working on the code for accomplishing different tasks (Kordaki 2012).

Results from similar research by Kalelioğlu and Gülbahar (2014) indicate that there were significant increases in the mean of the factor of "self- confidence in their problem solving ability" but programming did not cause any significant differences in problem solving skills. Authors state that a possible solution to this problem could be the support of students with different activities that require high-order thinking in order to help them develop problem solving skills.

We consider that students of both groups were involved only in some aspects of the problem solving process, but these aspects were not enough to reveal a significant difference in the problem solving skills. At the end of the course, the majority of the students of the control group persisted with habits based solely on mathematical equations, but in the experimental group a smaller percentage persisted in this habit. Students of the experimental group were more involved in processes like recognizing patterns, examining a broad span of information, capability of explaining the meaning of their own numerical solutions to the problems, while they indicated that they used scientific procedures such as qualitative analysis of the problem by making hypotheses and analysis of the result. However, despite their involvement in the afore mentioned problem solving process, it seems that this engagement was not enough in order to provide significant difference with the control group. On the contrary, students of the control group were less involved in the processes of problem solving and they insisted mainly on applying standard operational procedures to tackle the problem without being engagement in issues like abstraction and generalization. This does not mean that chemistry and electrology do not contain elements of CT, and we attribute the difference in the score of the experimental group (pre and post test) in the intervention.

The results of this study reject the hypothesis that computer programming would improve the general problem solving ability of upper high school students. Further studies need to examine the relationship between computer programming and problem skills in mathematics and if future efforts do not provide a link between computer programming

and improved solving ability, we need to justify the reasons for this or to remove from the integration of problem solving with computer programming.

Reasoning skills results

The third research question addressed in this study is "Does teaching computer programming have a significant difference on students' reasoning skills?"

A statistical analysis of the scores obtained by each group in the pre-test and in the post-test compared within the same group was conducted in order to be able to draw conclusions about whether and how the computer programming course affected or not the experimental group. Considering the results, there was no significant difference in the scores of the control group. For the experimental group, the results indicated that computer programming did elicit a statistical significant difference in students' reasoning skills.

The Mann–Whitney-U test was used to compare the difference between the two groups in the pre-test and in the post-test. The test indicates that the two groups did not differ significantly in the pre-test. In the post-test, the test showed that the difference between the score of the two groups is significant and thus computer programming elicited a significant difference in students' reasoning skills.

It came as no surprise that computer programming can affect students' reasoning skills. Many researchers have advocated in favour of the aforementioned finding. For instance, Tu and Johnson's (1990) study indicates that students who have completed all programming instruction lessons have significantly improved their logical reasoning skills. In addition, Gorman and Bourne (1983) found that computer programming is effective in augmenting the reasoning skills of students. Respectively, the study of Degelman et al. (1986) also supports the aforementioned research results. In addition, this study's results add to Fox and Farmer (2011) study, in which they found that their experimental group had higher reasoning score than their control group, but the difference was not significant. Another important finding of this study is that it is not the specific programming language that contributes to students' skills (since the course taught aims at the development of algorithms and not at learning a specific programming language). Lai and Yang (2011) conducted a study to investigate the effect of visualised programming on the learners' reasoning skills. Their sample was sixth graders who took a Scratch programming course over one semester. The researchers found out that there was no significant effect on logical reasoning skills.

One of the aims of the study was to highlight the importance of the reasoning skills and how computer programming could contribute to these skills in general. The other aims were to investigate the role of computer programming on students' problem solving skills and self-efficacy in mathematics. For this purpose, an intervention course was designed in which students from the experimental group took part in an integrated course of Mathematics and computer programming. The students, with their teachers' assistance, developed a software application in Matlab, which was related to their Mathematics curriculum and specifically to Statistics. The MSQL questionnaire was employed to test students' self-efficacy, while the problem solving was tested according to students' performance in a Mathematics test administered by their Math teacher and according to the Greek national examination standards.

The main findings that emerged from the analysis of students' questionnaires and tests are the following:

1. Students' self-efficacy score in Mathematics was significantly enhanced after the computer programming instruction course.
2. Students' problem solving score in Mathematics was significantly enhanced for students that participated in the computer programming course. The difference with the control group though was not statistically significant.
3. Students' reasoning skills in general were significantly enhanced by the computer programming course.

This study has highlighted the importance of computer programming in Mathematics. Considering all outcomes presented in this research study, one conclusion that can be drawn is that computer programming may provide many benefits for students' cognitive skills that can be applicable to many courses and not only to this particular computing course. This designates that learning computing programming may have an impact on cultivating students' skills that can be transferred to other courses like Mathematics, Engineering and Science. We consider that further research is necessary in order to explore whether computer programming curriculum, combined with proper instructional methods, could have an impact on students' problem solving skills. In such a case, learning activities should be designed to include the dimensions of computational thinking before students start programming in order to solve an authentic problem.

References

Ali, R., Akhter, A., Shahzad, S., Sultana, N., & Ramzan, M. (2011). The impact of motivation on students' academic achievement in problem based learning environment. *International Journal of Academic Research*, *3*, 306–309. www.ijar.lit.az.

Artino, A.R. (2006). A review of the motivated strategies for learning questionnaire. resource document. University of Connecticut. http://www.sp.uconn.edu/~aja05001/comps/documents/MSLQ_Artino.pdf. Accessed 12 March 2014.

Aukrust, V. G. (2011). *Learning and cognition in education*. Oxford: Elsevier.

Bandura, A. (1994). Self-efficacy. In V. S. Ramachaudran (Ed.), *Encyclopedia of human behavior* (pp. 71–81). New York: Academic.

Bandura, A., Barbaranelli, C., Caprara, G. V., & Pastorelli, C. (1996). Multifaceted impact of self-efficacy beliefs on academic functioning. *Child Development, 67*(3), 1206–1222.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48–54.

Bescherer, C., & Zimmermann, M. (2013). *Learning mathematics using ICT in pre-service teacher education* (pp. 142–151).

Blessing, L. T., & Chakrabarti, A. (2009). *DRM, a design research methodology*. London: Springer.

Bruner, J.S. (Ed.) (1957). Going beyond the information given. In Contemporary approaches to cognition: *A symposium held at the University of Colorado* (pp. 41–69). Cambridge: Harvard University Press.

Caprara, G. V., Vecchione, M., Alessandri, G., Gebrino, M., & Barbaranelli, C. (2011). The contribution of personality traits and self-efficacy beliefs to academic achievement: A longitudinal study. *British Journal of Educational Psychology, 81*(1), 78–96.

Caswell, R., & Nisbet, S. (2005). Enhancing mathematical understanding through self-assessment and self-regulation of learning: The value of meta-awareness. In P. Clarkson., A. Downton, D. Gronn, M. Horne, A. McDonough, R. Pierce, & A. Roche (Eds.), Building connections: Research, theory and practice. *Proceedings of the 28th annual conference of the Mathematics Education Group of Australasia, Melbourne*, 1(1), 209–216. Sydney: MERGA.

Clement, J., Lochhead, J., & Soloway, E. (1990). Positive effects of computer programming on students' understanding of variables and equations. *ACM'80 Proceedings of the ACM 1980 annual conference* (pp. 467–474), New York.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology, 76*(1), 1051–1058.

Cortina, T. (2007). An introduction to computer science for non-majors using principles of computation. *Proceedings of the 38th SIGCSE technical symposium on computer science education* (pp. 218–222). doi:10.1145/1227310.1227387.

Dalton, D., & Goodrum, D. (1991). The effects of computer programming on problem-solving skills and attitudes. *Journal of Educational Computing Research, 7*(4), 483–506.

Degelman, D., Free, J. U., Scarlato, M., Blackburn, J. M., Golden, T., & Colltye, E. N. (1986). Concept learning in preschool children: Effects of a short-term logo experience. *Journal of Educational Computing Research, 2*(2), 199–205.

Denning, P. J. (2009). Beyond computational thinking. *Communications of the ACM, 52*(6), 28–30.

DiSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. Cambridge: MIT Press.

Doubé, W., & Lang, C. (2012). Gender and stereotypes in motivation to study computer programming for careers in multimedia. *Computer Science Education, 22*(1), 63–78.

Einhorn, S. (2012). *Micro-worlds, computational thinking, and 21st century learnin*. White Paper: Logo Computer Systems Inc.

English, L. D., & Halford, G. S. (2012). *Mathematics education: Models and processes*. New York: Routledge.

Ennis, D. L. (1994). Computing, problem-solving instruction and programming instruction to increase the problem-solving ability of high school students. *Journal of Research on Computing in Education, 26*(4), 489–496.

Ennis, R. H., Millman, J., & Tomko, T. N. (1985). *Cornell critical thinking tests level x and level z manual*. Pacific Grove: Midwest Publications Critical Thinking Press.

Ennis, R. H., & Paulus, D. (1965). *Critical thinking readiness in grades 1-12: Phase 1: deductive reasoning in adolescence*. Ithaca: Cornell University.

Falkner, K., & Palmer, E. (2009). Developing authentic problem solving skills in introductory computing classes. *ACM SIGCSE Bulletin, 41*(1), 4–8.

Farrokhi, F. (2012). Rethinking convenience sampling: Defining quality criteria. *Theory and Practice in Language Studies, 2*(4), 784–792.

Fife-Schaw, C. (2000). Quasi-experimental designs. In G. M. Breakwell, J. A. Smith, & D. B. Wright (Eds.), *Research methods in psychology* (pp. 74–87). California: SAGE Publications Ltd.

Foshay, R., Kirkley, J. (2003). Principles for teaching problem solving. Technical paper 4. Resource document. PLATO Learning, Inc. http://files.eric.ed.gov/fulltext/ED464604.pdf. Accessed Jan 2015.

Fox, R. W., & Farmer, M. E. (2011). The effect of computer programming education on the reasoning skills of high school students. In H. R. Arabnia, V. A. Clinsy, & L. Deligiannidis (Eds.), *Proceedings of the international conference on frontiers in education: Computer science and computer engineering (FECS'11* (pp. 187–193). USA: CSREA Press.

Gandhi, H., & Varma, M. (2009). Strategic content learning approach to promote self-regulated learning in mathematics. *Proceedings of epiSTEME, 3,* 119–124.

Gorman, H., & Bourne, L. (1983). Learning to think by learning logo: Rule learning in third-grade computer programmers. *Bulletin of the Psychonomic Society, 21*(3), 165–167.

Govender, I. (2007). Experiences of learning and teaching: Problem solving in computer programming. *African Journal of Research In Mathematics, Science and Technology Education, 11*(2), 39–50.

Green, A. J. K., & Gilhooly, K. (2005). Problem solving. In N. Braisby & A. Gellatly (Eds.), *Cognitive psychology*. Milton Keynes: Open University Press.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43. doi:10.3102/0013189X12463051.

Guzdial, M. (2008). Paving the way for computational thinking. *Communications of the ACM, 51*(8), 25–27.

Güzeller, C. O., & Akin, A. (2012). The effect of web-based mathematics instruction on mathematics achievement, attitudes, anxiety and self-efficacy of 6th grade students. *International Journal of Academic Research in Progressive Education and Development, 1*(2), 42–54.

Halpern, D. F. (2003). *Thought and knowledge: An introduction to critical thinking* (4th ed.). Mahwah: Lawrence Erlbaum Associates.

Howe, J. A. M., Ross, P. M., Johnson, K. R., Plane, F., & Inglis, R. (1989). Teaching mathematics through programming in the classroom. In E. Soloway & J. C. Spohrer (Eds.), *Studying the novice programmer* (pp. 43–55). Hillsdale: Lawrence Erlbaum.

Jonassen, D. H. (2003). Using cognitive tools to represent problems. *Journal of Research on Education on Technology, 35*(3), 362–381.

Jonassen, D.H. and Reeves, T.C. (2001). Learning with technology: Using computers as cognitive tools, In D.H. Jonassen (Ed), *Handbook of research for educational communications and technology*. Resource Document. http://www.aect.org/edtech/ed1/24/index.html. Accessed May 2013.

Jones, S.P. (2011). Computing at School, international comparisons. Resource Document http://www.csta.acm.org/About/sub/AboutFiles/IntlComparisonsv5.pdf. Accessed Dec 2013.
Kagan, D. M. (2006). Research on computer programming as a cognitive activity: implications for the study of classroom teaching. *Journal of Education for Teaching: International research and pedagogy, 15*(3), 177–189.
Kalelioğlu, F., & Gülbahar, Y. (2014). The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33–50.
Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2011). Understanding computational thinking before programming: developing guidelines for the design of games to learn introductory programming through game-play. *International Journal of Game-Based Learning, 1*(3), 30–52.
Kordaki, M. (2012). Diverse categories of programming learning activities could be performed within Scratch. *Procedia-Social and Behavioral Sciences, 46,* 1162–1166.
Lai, A., & Yang, S. (2011). The learning effect of visualized programming learning on sixth graders' problem solving and logical reasoning abilities. In: International conference on electrical and control engineering (ICECE), 16–18 Sept 2011, 6940–6944. Yichang.
Lajoie, S. P. (1993). Computer environments as cognitive tools for enhancing learning. In S. P. Lajoie & S. J. Derry (Eds.), *Computers as cognitive tools* (pp. 261–288). Hillsdale: Lawrence Erlbaum Associates Inc.
Lavonen, J. M., Meisaloa, V. P., Lattua, M., & Sutinenb, E. (2003). Concretising the programming task: A case study in a secondary school. *Computers & Education, 40*(2), 115–135.
Leopold, C., & Leutner, D. (2015). Improving students' science text comprehension through metacognitive self-regulation when applying learning strategies. *Metacognition Learning, 10,* 313–346. doi:10.1007/s11409-014-9130-2.
Lesh, R., & Zawojewski, J. (2007). Problem-solving and modeling. In F. Lester (Ed.), *Second handbook of research on mathematics teaching and learning* (pp. 763–804). Reston: NCTM.
Linn, M. C. (1985). The cognitive consequences of programming instruction in classrooms. *Educational Researcher, 14*(5), 14–29.
Lohman, D. F., & Lakin, J. M. (2009). Reasoning and Intelligence. In R. J. Sternberg & S. B. Kaufman (Eds.), *Handbook of intelligence* (2nd ed., p. 183). New York: Cambridge University Press.
Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., & Resnick, M. (2004). Scratch: a sneak preview. *Second International Conference on Creating, Connecting, and Collaborating through Computing* (pp. 104–109), Kyoto, Japan.
Matlin, M. (2005). *Cognition* (6th ed.). Hoboken: Wiley.
McLellan, J. (2009). Establishing a benchmark for the deductive reasoning abilities of United Arab Emirates University Business students, learning and teaching in higher education. *Gulf persectives, 6*(2), 1–16.
Michalewicz, Z., & Michalewicz, M. (2008). *Puzzle-based learning: An introduction to critical thinking, mathematics, and problem solving*. Victoria: Hybrid Publishers Pty Ltd.
Miller, R. B., Kelly, G. N., & Kelly, J. T. (1988). Effects of logo computer programming experience on problem solving and spatial relations ability. *Contemporary Educational Psychology, 13*(4), 348–357.
Millsap, R. E., & Maydeu-Olivares, A. (2009). *The SAGE handbook of quantitative methods in psychology*. Thousand Oaks: SAGE.
OECD. (2013). *Problem solving framework, in PISA 2012, assesment and analytical framework: mathematics, reading, science*. Problem Solving and Financial Literacy: OECD Publishing.
Pajares, F., & Urdan, T. C. (2005). *Self-efficacy beliefs of adolescents*. Greenwich: Information Age.
Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. New York: Basic Books Inc.
Pardamean, B., Honni, H., & Evelin, E. (2011). The effect of logo programming language for creativity and problem solving. *Proceedings of the 10th WSEAS international conference on E-Activities. World Scientific and Engineering Academy and Society (WSEAS)*, 151–156.
Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas Psychology, 2*(2), 137–168.
Pea, R. D., Kurland, D. M., & Hawkins, J. (1985). Logo and the development of thinking skills. In M. Chen & W. Paisley (Eds.), *Children and microcomputers: Research on the newest medium* (pp. 193–317). Norwood: Ablex Publishing Corp.
Pintrich, P. R., Smith, D.A.F., Garcia, T., & McKeachie, W. J. (1991). A manual for the use of the motivated strategies for learning questionnaire (MSLQ). Resource Document. Ann Arbor: University of Michigan, National Center for Research to Improve Postsecondary Teaching and Learning. http://files.eric.ed.gov/fulltext/ED338122.pdf. Accessed 12 Mar 2014.
Psycharis, S. (2016). The impact of computational experiment and formative assessment in inquiry-based teaching and learning approach in STEM education. *Journal of Science Education and Technology, 25*(2), 316–326.

Robbins, J. K. (2011). Problem solving, reasoning, and analytical thinking in a classroom environment. *The Behaviour Analyst Today, 12*(1), 40–47.

Saeli, M., Perrenet, J., Wim, M. G., Joochems, W., & Zwaneveld, B. (2011). Teaching programming in secondary school: A pedagogical content knowledge perspective. *Informatics in Education, 10*(1), 73–88.

Schmidt, H. G., Rotgans, J. I., & Yew, E. H. J. (2011). The process of problem-based learning: What works and why. *Medical Education, 45,* 792–806. doi:10.1111/j.1365-2923.2011.04035.x.

Schoenfeld, A. (1983). The wild, wild, wild, wild world of problem solving: A review of sorts. *For the Learning of Mathematics, 3,* 40–47.

Schunk, D. (1989). Social cognitive theory and self regulated learning. In B. Zimmerman & D. Schunk (Eds.), *Self-regulated learning and academic achievement: Progress in cognitive development research* (pp. 83–110). New York: Springer.

Sendov, B., & Sendova, E. (1995). East or West-GEOMLAND is best, or does the answer depend on the question? In A. diSessa, C. Hoyles, & R. Noss (Eds.), *Computers and exploratory learning* (pp. 59–78). Berlin: Springer.

Soloway, E. (1993). Should we teach students to program? *ACM Communications, 36,* 21–24.

Spector, J. M. (2003). Problems with problem-based learning: Comments on model-centered learning and instruction in seel. *Technology, Instruction, Cognition and Learning, 1*(4), 359–374.

Stanic, G., & Kilpatrick, J. (1988). Historical perspectives on problem solving in the mathematics curriculum. In R. I. Charles & E. A. Silver (Eds.), *The teaching and assessing of mathematical problem solving* (pp. 1–22). Reston: National Council of Teachers of Mathematics.

Sternberg, R. J. (2008). *Cognitive psychology*. Belmont: Cengage learning.

Subrahmanyam, K., Kraut, R. E., Greenfield, P. M., & Gross, E. F. (2000). Computer use on children's activities and development, the future of children. *Children and Computer Technology, 10*(2), 123–144.

Syslo, M.M., Kwiatkowska, A.B. (2006). Contribution of informatics education to mathematics education in schools. In R.T., Mittermeir (Ed.), *LNCS: 2nd International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP, Proceedings* (Vol. 4226, pp. 209–219), Vilnius: Springer.

Szlávi, P. & Zsakó, L. (2006). Programming versus application. In R.T., Mittermeir (Ed.), *LNCS: 2nd International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP, Proceedings* (Vol. 4226, pp. 48–58). Vilnius: Springer.

Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia Social and Behavioral Sciences, 8*(1), 561–570.

Tu, J. J., & Johnson, J. R. (1990). Can computer programming improve problem-solving ability. *ACM SIGCSE Bulletin, 22*(2), 30–37.

Unuakhalu, M. (2008). Enhancing problem-solving capabilities using object-oriented programming language. *Journal of Educational Technology Systems., 37*(2), 121–137.

VanLengen, C. A., & Maddux, C. D. (1990). Does instruction in computer programming improve problem solving ability? *Journal of Information Systems Education, 2*(2), 11–16.

Wells, D. (2012). Computing in schools: Time to move beyond ICT? *Research in Secondary Teacher Education, 2*(1), 8–13.

White, G. L. (2011). Visual basic programming impact on cognitive style of college students: need for prerequisites. *Information Systems Education Journal, 10*(4), 74–83.

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

Woosley, S. (2012). Chapter 7: Using Tests. Resource Document. http://cms.bsu.edu//media/WWW/DepartmentalContent/Effectiveness/pdfs/Wkbk/WBKM12012%20%20Ch%207.pdf. Accessed 15 May 2013.