An Experience Report on the Effectiveness of Five Themed Workshops at Inspiring High School Students to Learn Coding

Nearchos Paspallis, Irene Polycarpou, Panayiotis Andreou, Josephina Antoniou, Paris Kaimakis,
Marios Raspopoulos, Maria Terzi
University of Central Lancashire—Cyprus
Larnaca, Cyprus
{npaspallis,ipolycarpou,pgandreou,jantoniou,pkaimakis,mraspopoulos,mterzi}@uclan.ac.uk

ABSTRACT

Today there is a high demand for computing programmers, and at the same time a shortage of skilled professionals. This has triggered the creation of many initiatives in the past few years, with the aim of reversing the phenomenon. To achieve this, such events are designed to promote a more appealing image for programming, both as a profession and as a skill. This paper describes one such initiative, which uses a unique blend of differently themed, parallel workshops to motivate high school students to learn programming. With the use of questionnaires, we survey the participants and present our findings concerning the effectiveness of these workshops to engage the participants, to promote the value of coding, and to encourage the participants to consider a career in the field. We evaluate our results both at a general level, as well as by comparison among five individually themed workshops.

CCS CONCEPTS

• **Social and professional topics** → **Computing literacy**; *Computational thinking*; *Computer science education*;

KEYWORDS

Computer Science Education, Gamification, Coding

1 INTRODUCTION

With the advancement of technology and its penetration in almost every aspect of modern life, programming has become an indispensable skill in a wide range of professions. As a result, there is a high demand for trained computer programmers. At the same time, we observe a shortage of skilled programmers, and it is projected

that this gap will widen even further. One of the main factors contributing to this shortage is lack of motivation of young people to learn programming and pursue related studies and careers.

Learning to code is nowadays considered a skill of high importance, like knowing how to read, write and do basic mathematics [16]. Many initiatives, like the Code Week in Europe [8] and the Hour of Code in the United States [15], aim to emphasize the importance of coding and inspire the youth to invest more time and effort in learning to code. In 2013, we endeavored to set up a similarly themed national event to promote the value of coding. This event, titled *Code Cyprus*, is a one-day, annual event aiming to "*inspire high school students to take an interest in programming*". Code Cyprus includes multiple activities, competitions and prizes with the goal of engaging and inspiring the participants.

This event is designed to resemble a festival rather than a conference: its agenda includes paper-free registration, colorful name badges, a plenary motivation talk, and playful hands-on workshops. Its capstone is a gamified treasure hunt using mobile devices. The questions/hints are programming-related and the players are further incentivized via appealing prizes. This gamified activity aims primarily at further encouraging the participants to join the event, and also be engaged during the workshops [9].

The five workshops discussed in this paper employ different approaches to engage the participants, focusing on traditional programming languages, coding for games, robotics platforms, database query languages, and microelectronics. These themed and hands-on workshops share a common goal: guide the participants into understanding the fundamental concepts of *variables*, *conditionals* and *loops*. As the workshops take place in parallel, each participant attends exactly one of them. This has inspired us to perform a comparative study of the five workshops, to study and asses their effectiveness in engaging the participants, as well as in communicating the targeted programming concepts.

The rest of this paper is organized as follows: section 2 describes the five themed workshops and their individual approach in motivating the participants and explaining the core programming concepts. Section 3 presents the data collection method, as well as the design of the questionnaires. The results are then presented in section 4, followed by a discussion of related work in section 5. The paper closes with conclusions in section 6.

2 THE THEMED WORKSHOPS

The approach described in this paper builds on five independent workshops, all aiming to engage the participants and communicate the core programming concepts of *variables*, *conditionals* and *loops*. The workshops assumed different skill-sets of the participants. The

difficulty and themes were *beginners* (with programming in *Python*), intermediate (with *computer games* and *robotics*) and advanced (with *database query languages* and *microelectronics*). Each workshop offered its own unique setup and theme.

2.1 Beginners programming

The Beginners workshop aimed at newbies and had as its goal to explain the core programming concepts but also to cover additional skills on decision-making and designing. It used the Python programming language as the preferred medium for conveying coding principles to beginners. Python was selected as it has a smooth learning curve, allowing students to experiment with it easily, while at the same time not being too simplistic. The use of an interpreter to give the students immediate feedback to their coding attempts, as well as the use of a graphics library to allow the students to visualize their code, were elements that contributed towards an engaging workshop.

The workshop began with an introduction of data types through numeric and string examples. The manipulation of strings and numbers, two concepts familiar to young students from the real world, naturally led to the definition and use of variables, which in Python is a straightforward process and does not use additional concepts, such as the need to precede the use of a variable with a definition of its data type. The workshop continued with similar manipulation of strings and numbers all the way to conditionals and loops. Using examples that the students were already comfortable with, has helped to transition from the use of variables to the use of conditionals and loops. Once all three concepts (*i.e.*, variables, conditionals and loops) were presented and practiced, the workshop closed with a graphical exercise, which guided the students to visualize the effect of each of their coding statements as soon as it was executed.

2.2 Games challenge

This challenge was designed with the assumption that the attendants would have some prior knowledge of programming. For this reason, some time was dedicated to ensuring that learners understood the core concepts of programming.

The notion of variables was introduced by way of the idea of a box into which the programmer may place useful information (a number, a string, a Boolean, etc.), for use later in the program. The idea of the box was further reinforced by the instructor, as he performed gestures pretending to open the box and retrieving the information he had originally placed inside. The notions of conditionals and loops were described by means of examples, verbally, and through video illustrations. All three concepts were then brought together by studying simple code-blocks (using *Scratch* [12]) and by posing questions about the content of certain variables when the code-block had finished executing. A set of simple programming puzzles was then solved through the dialectic method, i.e. encouraging discussion to establish a commonly accepted solution.

Next the learners were presented with the problem of escaping from a labyrinth[1]. Everybody was impressed when it was announced that one can escape from any labyrinth by consistently

---

[1]A *simply-connected labyrinth*, to be precise, *i.e.*, a labyrinth consisting of walls which are connected to its outer boundary.

maintaining contact with the left- (or right-) hand-side wall. After this idea was digested in terms of a cartoon demonstration, the learners helped the instructor derive an algorithm, in terms of pseudo-code (involving variables, conditionals, and loops), for escaping from the labyrinth [5].

Finally, the ideas of variables, conditionals and loops were further reinforced by learning how to build (parts of) a game. For this task the Python programming language was used for its simplicity, alongside the *Code Combat* framework [3]. This stage was designed in a way that creates a competitive atmosphere among the learners, by encouraging quick thinking while maintaining the fun aspects of game development.

2.3 Robotics challenge

The aim of the Robotics Challenge is to engage the participants through a task where they need to program a robot to perform specific tasks. The Engino Robotic Platform (ERP) [4] was used in this workshop. This platform is specifically designed for primary and secondary education and it includes the basic components for building and programming a Robot, such as a controller, cables, touch and infrared sensors, motors and LEDs. The platform comes with a proprietary programming software which contains a library of functions using a visual, block-based programming language, including logical and control functions such as conditionals, loops and subroutines.

This was aimed to be an intermediate-level workshop, where students were assumed to have some basic knowledge about programming which mainly includes a basic understanding of variables, conditionals and loops. Nevertheless, these concepts were presented to the participants via some real-life examples.

The main activity included the programming of the ERP robot using two infrared (IR) sensors, so that it would follow a black line (line following challenge). The participants had to apply their knowledge on conditionals to trigger events when the IR sensors were activated, indicating that the robot is going off-course and thereafter apply the necessary corrections. Obviously, this was a continuous process therefore the students also had to apply their knowledge on loops using the WHILE mechanism. Additional programming challenges were given to students who managed to complete this objective early. The extra challenge included programming two colored LEDs (one on the left side and one on the right) to turn on and off when the robot turned left or right respectively.

2.4 Querying data with SQL

The objective of this advanced level workshop was to offer high school students an introduction to querying data from relational database management systems using SQL, in a fun and intuitive way. More specifically, the participating students were introduced to the most common operations of the SELECT statement by interacting with a real database composed of all the UEFA Champions League football matches from a particular season.

The workshop adopted an incremental approach to developing SQL SELECT queries by first introducing the FROM clause and then gradually exposing students to comparison predicates (WHERE), eliminating duplicates (DISTINCT), sorting (ORDER BY), aggregated functions (COUNT, MIN, MAX) and SQL scripting (using

T-SQL). The clauses were presented along with examples illustrated them. This was followed by a detailed description of the syntax and more query examples that showcased the clause's distinct features. Finally, using a snapshot of one of the table's data, the students got an opportunity to reflect on their understanding of the clause via hands-on practicing on several query problems.

The overall assessment of learning in this workshop comes after all clauses have been introduced and practiced upon. Since it has been argued that paper-based standardized testing approaches for facilitating the learning of SQL at higher education are sub-optimal and are partially responsible for decreasing student engagement [7], [11], this workshop employed an interactive web application, which first provided educators with tools to seamlessly construct SQL problems and deploy them; second, it allowed students to attempt to solve the problems and generated targeted feedback for any attempt; and, third utilized a leader-board, which displayed the number of correct answers per student, and thus increased engagement by encouraging them to compete.

2.5 Programming microelectronics with Arduino

This workshop focuses on communicating the core concepts of programming (*i.e.*, variables, conditions, loops) in a fun, interactive and engaging way using microelectronics programming with Arduino. The latter is a small-sized, low-cost, easy-to-use and scalable platform with various applications in engineering projects, but also in engineering education and student motivation [10, 17].

Arduino consists of a programmable circuit board and an IDE (Integrated Development Environment) which is used to write and upload sketches (high-level C++ programs) to the board via a USB cable. The Arduino boards come with input and output pins for connecting sensors (*e.g.*, thermostats, motion detectors) and actuators (*e.g.*, LEDs and motors), therefore enabling interaction with the environment. This workshop was designed so that students could experience a hands-on approach to programming the Arduino platform. The workshop was themed around the task of realizing a simple traffic lights system.

The workshop was organized in two parts. The first one focused on familiarizing students with programming, as well as with the Arduino IDE and the board itself. It provided a brief introduction to programming and used simple examples of how variables, conditions and loops are used in an Arduino project. For example, a variable was communicated as *the way the Arduino can remember the value of the pin a sensor is using* and the for-loop as something that can *turn on and off an LED in repeat*. Additionally, the first part of the workshop provided some basic circuit theory and it discussed the role of a breadboard in a multiple-elements project, the procedure of connecting sensors and actuators to the board via the breadboard, and how the IDE is used for writing and uploading sketches to the board.

The second part consisted of a hands-on implementation of a traffic-lights system using the Arduino, a breadboard, wires, and three colored LEDs. During this part, the students were first provided with written instructions on how to connect one LED light, and then how to write and upload the code that initialized the LED to the right pin and make it turn on and off with some delay. After

this, the students were instructed to use the other two LED lights, more wires and the knowledge acquired from the previous parts to build a complete traffic light system. Upon success, the students were further challenged to make the lights blink faster after the system had run for three times.

3 DATA COLLECTION AND QUESTIONNAIRE DESIGN

Data collection was planned so it could take place in a uniform manner in each of the five parallel workshops. The following subsections describe the data collection approach along with the design of a questionnaire used for that purpose.

3.1 Data collection

The data collection was organized around two phases:

- First, parts A and B of the designed questionnaire were completed by the participants at the beginning of the workshop to identify the existing knowledge and skills, as well as the motivation of each participant.
- Second, parts C and D were completed after the completion of the workshop. These parts aimed at identifying the possible improvement in coding skills of the participants, as well as the degree to which they motivate participants to invest more time into learning programming.

The data collection was organized and implemented with paper-based questionnaires. This is despite the availability of computers in four of five workshops, as well as the fact that transcription is generally harder and more expensive for paper-based feedback. Our choice was guided by the following: First, in most workshops computers were shared by two or three participants. This implied that choosing electronic forms would limit our ability to survey the maximum number of participants. It would also make it more difficult to ensure that the collected data was indeed personal. Second, as our questionnaire was split into parts A and B—to be completed before the workshop—and parts C and D—to be completed after the workshop—using paper-based questionnaires has made it easier to ensure that the same persons completed the right questionnaire.

3.2 Questionnaire design

The questionnaire is designed to include both demographic and programming-specific questions. In terms of programming, it shows questions using Blockly, which is "*a type of visual block language development kit that allows the rapid construction of new block-based visual programming languages to address a specific pedagogical or content focus*" [13]. Furthermore, Blockly is related and very similar to Scratch [12], which was familiar to some of the participants.

The questionnaires were organized in four sections as follows:

- A This section collects the year of birth, and gender of the participant. Also, it asks the participant to perform a self-assessment of their skills in mathematics and any existing skills in programming. Finally, it collects the participant's perception of programming as a skill and career choice, and its potential to help them in their future career.
- B This next section provides three Blockly-based programming questions, one for each of the core programming concepts of

Figure 1: Sample page from the questionnaire (Part B - English version).

variables, *conditionals*, and *loops*. An excerpt of the English version of Part B is illustrated in Fig. 1.

C Similar to the previous section, this section includes Blockly-based programming questions, again one for each of the core programming concepts of *variables*, *conditionals*, and *loops*. These are symmetric to their corresponding questions of Part B, both in terms of semantics as well as in terms of complexity. Unlike Part B though, the questions in this section are answered after the completion of the workshop.

D The final section of the questionnaire asks the participants to perform a self-assessment of their development over the workshop. They are asked to report whether they think their programming skills were improved in general, as well as to rate which of the three core programming concepts they understood the least and which one the most. Finally, they are asked to provide their personal view about programming *after* their workshop, as well as whether they would consider a future career in computing.

The questionnaires included a total of 18 multiple choice questions, 9 in Parts A and B (*i.e.*, before the workshop) and 9 in Parts C and D (*i.e.*, after the workshop). The questions were designed so

that the questionnaire could be completed in a relatively short time, with the aim of maximizing the completion ratio. Finally, as the workshops included both Greek and English speaking high school students, we made the questionnaires available in both languages. Participants were given the choice to use the questionnaire of their preferred language.

4 RESULTS AND LESSONS LEARNED

The data collection took place on March 4th, 2017 as part of Code Cyprus 2017. In each of the five workshops, participants were asked to complete Parts A and B first. Then the workshops proceeded as described in section 2, with the students eventually asked to complete Parts C and D at the end of the workshops. From a total of 112 attendees, 106 of them completed and submitted a questionnaire and out of those, 99 were retained (7 were dropped because they were incomplete).

4.1 Demographic data

The event was advertised nationally with school visits, participation to science festivals, and online via its website and Facebook page. The participants registered themselves to their preferred workshop ahead of the event. Naturally, each workshop offered a limited number of slots, a constraint imposed by physical capacity limitations. Consequently, in some cases participants were not able to select their first choice (*i.e.*, when the availability of their preferred workshop was exhausted), as was the case with the Robotics and the Beginners workshops. The demographic data for the 99 participants who submitted a valid questionnaire are summarized in Table 1. These are presented both grouped by workshop and overall.

The participants were predominantly male (~63%), confirming the existing gender gap in STEM [1]. Notably, female participation to the topics of SQL and Arduino was zero. At the same time, the Beginners and Robotics workshops attracted a more balanced ratio, with 53% and 45% female participation respectively.

Table 1: Summary of participants - demographic data

Workshop	Ard.	Beg.	Gam.	Rob.	SQL	Total
Cohort	8	34	18	29	10	*99*
Gender						
Male	8	16	12	16	10	*62*
Female	0	18	5	13	0	*36*
Other	0	0	1	0	0	*1*
Age						
Min	14	10	11	12	15	*10*
Median	15	14	15	14	15	*14*
Max	17	16	18	18	21	*21*

4.2 Participant skills

With the purpose of assessing the student confidence in programming, as well as its possible relation to workshop choice, we asked the participants to answer the following: "*How do you rate your skills in mathematics?*" and, "*How do you rate your skills in programming?*" In both questions, the students answered on a Likert scale, where the options were: 1 ("minimal"), 2 ("poor"), 3 ("average"), 4
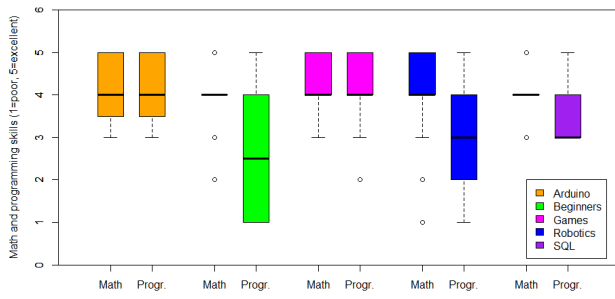
Figure 2: Boxplot of participant self-rated math and programming skills, grouped by workshop.

Figure 3: Boxplot of participants' view of programming, grouped by workshop.

("good") and 5 ("excellent"). Based on the received feedback, we have generated the boxplots shown in Figure 2.

In these, we observe that the participants who chose the themed workshops of Arduino and Games were generally confident in both their math and programming skills, with most of them rating their skills as "good" or "excellent". The participants in the other three workshops appeared also as mostly confident in their math skills (with a median of "good") but were less confident in their programming skills (with a median of "average"). This is reasonable, especially in the context of the Beginners workshop which was presented as the natural choice for programming novices. However, a noteworthy insight is that participants with limited programming skills seemed to be interested in specially-themed workshops, like the SQL, and the Robotics ones. This confirms the potential of robotics at inspiring and encouraging learning to code, as it has also been argued in the literature (*e.g.*, by Barreto and Benitti [2]).

4.3 Participants' view of programming

As the main mission of the event was to "*encourage high school students to take an interest in programming*", the questionnaire also aimed to collect the participants' views towards programming. More specifically, it asked the participants to rate their perception of programming before and after each workshop, by selecting one of the options: 1 ("indifferent"), 2 ("interested") and 3 ("excited").

Naturally, participants to the event cannot be assumed to be a truly uniform sample of the general population, as they all had to find the event and *voluntarily* register for it. This is reflected in Figure 3, where it is shown that most participants had already a very positive opinion of programming (most of them were "excited" with it). The only exception was with the participants registered to attend the Games-themed workshop (median was "interested"). Notably, the already high opinion improved even further after the workshops, with the Games participants increasing their median threshold to also "excited".

These results are supported by the fact that most participants joined the event because they "love programming", as shown in Figure 3. While the positive attitude of the participants is certainly encouraging, it also shows that this kind of event might be facing a significant challenge in terms of reaching the *right* audience. More specifically, this showcases the challenge that such events are often attended by those who are already positively biased towards
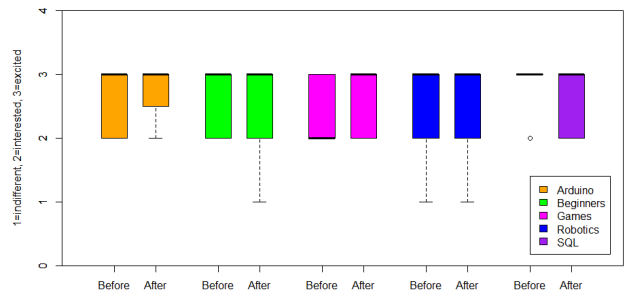
learning to program. In many cases participants even know programming beforehand. It can be argued then, that those who would benefit the most out of these events are persons who have either not had the opportunity to get a complete picture of what programming is or, worse, they have mistakenly developed a negative picture for it. How to reach out to and engage this group though, remains an open and challenging problem.

4.4 Participants' understanding of programming concepts

As mentioned in section 2, the educational aspects of the workshops were centered on the three core concepts of *variables*, *conditionals*, and *loops*. After the completion of the workshops, the participants were asked to identify the most and the least understood of these concepts. Their responses are summarized in Table 2. Arguably, the least understood concept was that of variables, which is probably the concept requiring the most effort in terms of abstract thinking, and at the same time one of the core capacities needed to develop coding skills [6].

Table 2: Summary of most and least understood programming concepts as reported by the participants

	Variables	Conditionals	Loops	All	Nothing	Other	Empty
Most understood	21	30	37	2	3	2	4
Least understood	31	24	23	0	10	2	9

Furthermore, the participants were assessed indirectly via the questionnaire and specifically its programming questions in sections B and C. The results of these are summarized in Figure 4. The findings of this experiment are rather inconclusive, as we can observe that in some cases (Beginners and Games workshops) there was some improvement (*i.e.*, participants scored *higher* after the workshop), in one case (Beginners) there was no change, and in the rest (Robotics and SQL) we observe the results to worsen (*i.e.*, the participants scored *lower* after the workshops). While we cannot be sure, we hypothesize this to be attributed to two factors. First,
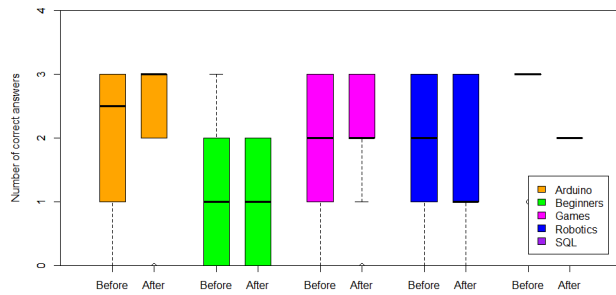
Figure 4: Comparison of correct answers before and after the intervention, grouped by workshop.

6 CONCLUSIONS

In this paper we have described and evaluated a novel approach for encouraging teenagers to take an interest in programming, using five specially-themed workshops. We found that this approach has provided some important insights: we have verified the gender gap evident in programming schools and professions, we have confirmed the value of themed activities, such as robotics, in motivating the students, and we have identified a challenge that relates to the difficulty of reaching an audience who could potentially become interested in programming. As this research is an ongoing work, in the future we will aim to collect more measurements—aiming to attract a more uniform set of participants—as well as qualitatively assess the participants perceptions via semi-structured interviews.

these were among the hardest workshops and it is likely that some participants selected them without having the necessary skills, or a choice for an easier one. Second, this could be attributed, to some extent, to the fact that the programming questions of section C (after the intervention) were a bit harder than those of section B (before the intervention). Either way, the *difference* was very small.

5 RELATED WORK

As a result of the demand/supply gap in computing jobs, many initiatives were launched aiming to improve the appeal of coding both as a profession and as a skill. This issue and the resulting actions (*e.g.*, events) have been studied by researchers with the aim of identifying the magnitude of the problem, as well as measuring the effectiveness of different approaches.

In Europe, Code Week was introduced as an *umbrella* for organizing and supporting multiple related events across the continent. Code Week is described and compared to similar activities by Moreno-Leòn and Robles [8]. Similarly, the Hour of Code is the pioneer event of its type taking place in the United States [15].

While the literature has very few works on explicitly aiming to measure or compare the effectiveness of individual events, like this paper does, it does include many studies with a focus on the factors that influence success in computer science. For instance Wilson [14] has identified that "*the participation of women in computer science from high school to graduate school diminishes at an alarming rate.*" Similarly to our study, the author examined *previous programming skills* as a factor influencing the participants' success. This study identified three factors able to predict success in computer science: First, and most importantly, the *comfort level* was found to contribute positively (which to some extent could be related to our measure of *existing programming skills*). Second, the student *math background* was found to also positively predict success in computer science (as in [6]), and was also explicitly asked in our study (self-assess existing math skills). Finally, the study found that *attribution to luck* had a negative influence to the determination of future success (something we have not explicitly studied). Additionally, the author identified significant gender differences in these factors, something that is reflected in our findings as well.

REFERENCES

[1] David Beede, Tiffany Julian, David Langdon, George McKittrick, Beethika Khan, and Mark Doms. 2011. Women in STEM: A Gender Gap to Innovation (ESA Issue Brief #04-11). https://eric.ed.gov/?id=ED523766. Accessed: 2017-11-10.
[2] Fabiane Barreto Vavassori Benitti. 2012. Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education* 58, 3 (2012), 978 – 988. https://doi.org/10.1016/j.compedu.2011.10.006
[3] Code Combat. 2017. Learn how to code by playing a game. http://codecombat.com/. Accessed: 2017-11-10.
[4] G. A. Demetriou, A. Lambrou, N. Eteokleous, and C. Sisamos. 2013. The Engino Robotics Platform (ERP) controller for education. In *21st Mediterranean Conference on Control and Automation*. 567–572. https://doi.org/10.1109/MED.2013.6608778
[5] Henry Ernest Dudeney. 1958. *Amusements in Mathematics*. Dover Publications.
[6] Jeff Kramer. 2007. Is Abstraction the Key to Computing? *Commun. ACM* 50, 4 (April 2007), 36–42. https://doi.org/10.1145/1232743.1232745
[7] Sudhir I. Mehta and Nem W. Schlecht. 1998. Computerized Assessment Technique for Large Classes. *Journal of Engineering Education* 87, 2 (1998), 167–172. https://doi.org/10.1002/j.2168-9830.1998.tb00337.x
[8] J. Moreno-León and G. Robles. 2015. The Europe Code Week (CodeEU) initiative shaping the skills of future engineers. In *2015 IEEE Global Engineering Education Conference (EDUCON)*. 561–566. https://doi.org/10.1109/EDUCON.2015.7096025
[9] Nearchos Paspallis. 2015. A Gamification Platform for Inspiring Young Students to Take an Interest in Coding. In *Information Systems Development: Transforming Organisations and Society through Information Systems (ISD2014 Proceedings)*. Varaždin, Croatia. http://aisel.aisnet.org/isd2014/proceedings/Education/1
[10] Raúl Pérula-Martínez, Juan Miguel García-Haro, Carlos Balaguer, and Miguel A. Salichs. 2016. Developing Educational Printable Robots to Motivate University Students Using Open Source Technologies. *Journal of Intelligent & Robotic Systems* 81, 1 (01 Jan 2016), 25–39. https://doi.org/10.1007/s10846-015-0205-3
[11] Julia Coleman Prior. 2003. Online Assessment of SQL Query Formulation Skills. In *Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20 (ACE '03)*. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 247–256. http://dl.acm.org/citation.cfm?id=858403.858433
[12] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. 2009. Scratch: Programming for All. *Commun. ACM* 52, 11 (Nov. 2009), 60–67. https://doi.org/10.1145/1592761.1592779
[13] Jake Trower and Jeff Gray. 2015. Blockly Language Creation and Applications: Visual Programming for Media Computation and Bluetooth Robotics Control. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE '15)*. ACM, New York, NY, USA, 5–5. https://doi.org/10.1145/2676723.2691871
[14] Brenda Cantwell Wilson. 2002. A Study of Factors Promoting Success in Computer Science Including Gender Differences. *Computer Science Education* 12, 1-2 (2002), 141–164. https://doi.org/10.1076/csed.12.1.141.8211 arXiv:http://dx.doi.org/10.1076/csed.12.1.141.8211
[15] Cameron Wilson. 2015. Hour of Code—a Record Year for Computer Science. *ACM Inroads* 6, 1 (Feb. 2015), 22–22. https://doi.org/10.1145/2723168
[16] Jeannette M. Wing. 2006. Computational Thinking. *Commun. ACM* 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215
[17] W. S. Yoo and S. A. Shaik. 2016. Development of Home Management System Using Arduino and AppInventor. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 2. 379–380. https://doi.org/10.1109/COMPSAC.2016.96