ORIGINAL RESEARCH
published: 08 February 2022
doi: 10.3389/fcomp.2022.824995

Game Design, Gender and Personalities in Programming Education

Anastasios Theodoropoulos and George Lepouras*

Human-Computer Interaction and Virtual Reality (HCI-VR) Laboratoty, University of Peloponnese, Tripoli, Greece

In a changing world programming learning is becoming more and more essential in education. And, there are many programming environments and teaching approaches that address the learning needs of students in CS education. A single programming tool or a method do not fit all students. Research has focused on gender differences and there is high interest in increasing female participation. Games and especially game-design tend to impact perceived usefulness of programming environments. Moreover, personality traits like cognitive style and emotional intelligence (EQ) seem to correlate with technology and achievement in programming. In this study, the effects of three different programming environments were investigated, in high school settings, by creating games and taking into account gender and personality characteristics. Three groups were formed, group A created games with Scratch, group B used App Inventor and made games for mobile devices, while group C created interactive stories-games with Alice 3D. This study was seeking to find possible biases based on gender, learning perception, usage and students' personalities between the three experimental conditions. One hundred and sixty three students aged 14–15 years old participated in the study, and data were collected through pre activity and post activity questionnaires. Results show different gender preferences for the three programming tools and, in some cases, different personalities (cognitive styles and EQ) have different learning preferences. Moreover, all programming environments had different emotional effects on the students. The study concludes with guidelines for programming learning environments that respect individual learning preferences and aim to maximize learning effectiveness.

Keywords: game design, games for learning, programming learning, personality traits, cognitive style, emotional intelligence

OPEN ACCESS

Edited by:
Leontios J. Hadjileontiadis, Khalifa University, United Arab Emirates

Reviewed by:
Styliani Kleanthous, Open University of Cyprus, Cyprus
Sherlock Licorish, University of Otago, New Zealand

****Correspondence:***
Anastasios Theodoropoulos
ttheodor@uop.gr

Specialty section:
This article was submitted to Human-Media Interaction, a section of the journal Frontiers in Computer Science

Received: *29 November 2021*
Accepted: *05 January 2022*
Published: *08 February 2022*

Citation:
Theodoropoulos A and Lepouras G (2022) Game Design, Gender and Personalities in Programming Education. Front. Comput. Sci. 4:824995. doi: 10.3389/fcomp.2022.824995

INTRODUCTION

Programming is the core characteristic in the field of Computer Science Education (CSE) (Ben-Ari, 1998). Literature shows that learning programming enhances Computational Thinking (CT) and problem-solving skills, both essential skills for the 21st century (Ben-Ari, 1998; Mathew et al., 2019). In a changing world, programming learning is becoming more and more central in many educational systems. In fact, in many countries (Nagashima et al., 2018; Fessakis and Prantsoudi, 2019; Qian et al., 2019), programming learning is a part of the curriculum, implying that vast numbers of children will be exposed to code as well as software and applications helping them to learn programming.

Programs comprise of instructions that command a computer/machine to behave in a certain way and can have several looks, such as text or blocks (Weintrop and Wilensky, 2017). Though, learning programming has been a challenging subject for students across the world. There are numerous difficulties that students face when learning the logic of programming (Gomes and Mendes, 2007; Bosse and Gerosa, 2017; Qian et al., 2019). In a broad sense, they face difficulties in understanding the basic structure and concepts of programming and when designing a program to solve certain tasks. Therefore, several tools and environments have been developed to assist them by creating applications that are related with their interests (Broll et al., 2017) and to make programming accessible to various groups (e.g., different ages) with no previous coding experience.

Indeed, programming learning environments address the learning needs of students (Matthews et al., 2009) and try to make the coding tasks more usable (Truong et al., 2003), make tutor feedback more effective and easily providable (Bancroft and Roe, 2006), and even allow students to spot coding errors easily (Nagashima et al., 2018). Moreover, such environments apply different means to engage users. For example, some use games and/or gamification to teach students (Shanahan, 2009) while others use visual tools (Broll et al., 2017). There are also studies focusing on student feedback (e.g., Matthews et al., 2009) and in general students prefer hands-on experiences when learning programming (Paul et al., 2006).

There are also cloud-based systems wishing to overcome distance and temporal constraints (Kiridoshi et al., 2018), mixed physical and digital programming environments [e.g., controlling robots (Ryokai et al., 2009)], environments that support object modeling (which are particularly easy for beginners) (Schwartz et al., 2006) and so on.

But how students perform when using different programming environments? Previous research highlights the need to use different learning environments under certain circumstances, depending on the students' needs (McKenna, 2004; Gunbatar and Karalar, 2018). The effectiveness of such learning environments and approaches is becoming the focus of current research efforts and different methods are used for this purpose. For example, an eye tracking study showed significant differences between kids and teens in coding activities. Kids were more interested in the appearance of the characters whereas teens were more interested in testing different hypotheses in relation to the code (Papavlasopoulou et al., 2017). Another study found important issues for non-native English-speaking children in programming activities, concluding that there is a need for simplified English, culturally-agnostic coding examples and embed inline dictionaries (Guo, 2018). Furthermore, it is also important to note that it is not only the programming learning environment that is important but also the teaching style followed, since it makes a big difference to allow active children involvement as opposed to more traditional, lecture style teaching (Makris et al., 2013). With the involvement of children in programming activities and the compulsory character these activities have in many educational systems worldwide, it seems necessary to proceed with clear evaluation plans (Von Hausswolff, 2017) in order to provide the necessary frameworks for the use of programming learning environments on a large scale (Hazzan et al., 2008).

Different pedagogical approaches are also applied to engage children with programming activities, like peer learning (Ndaiga and Salim, 2015), collaborative learning (Tholander et al., 2004) and game-based learning (Rugelj and Lapina, 2019). Kafai in her work (Kafai, 2012), presents game-design as a context for students' learning. Students from an elementary public school, programmed games to teach fractions to other students, so their classroom was transformed in a game studio for six months. The game-design approach became a medium for children's creativity and learning. A systematic review by Denner et al. (2019) shows that there are several educational benefits when designing and programming computer games especially in terms of learning and motivation. The current study focuses on three programming learning environments widely known and used (1) Scratch (Theodoropoulos et al., 2017); (2) App Inventor (Faria et al., 2010); and (3) Alice (Ivanović et al., 2014), in the perspective of game-design.

Furthermore, students have different preferences regarding coding assignments (Thomas et al., 2002). Important factors that influence their choices seem to be task dependent such as task difficulty, potential of enhanced learning experience and perceived benefit (Smith et al., 2014). However, there are other factors that influence their preferences which are task independent, like gender and personality. Gender is an important and well-recognized factor in STEM related fields and especially in programming education (McKenna, 2004) and there is a lack of studies exploring gender and programming environments in schools (Spieler and Slany, 2018). For example, Malik and Coldwell-Neilson (2018) adopted a four-stages model (approach, deployment, result, improvement) as a learning approach and indicated that female students performed better in some tasks while males performed better in others. In the same study, students' responses showed gender differences on satisfaction, regarding the learning approach that was followed. Another study, from Gunbatar and Karalar assess the effects of teaching programming with mBlock. The results showed that self-efficacy perceptions of boys toward programming were higher than the girls' before conducting the research, but this difference was smaller after the coding activities. However, games might be an effective way to battle gender stereotypes in programming education (Gunbatar and Karalar, 2018). Therefore, the first research question of this study is about gender and the programming learning environments:

RQ1: Is there any difference in students' gender and their preferences for creating games in different programming environments?

Rationale: Identify if students find some programming tools easier than others in terms of usability and learning, in correlation with their gender.

Moreover, previous research highlights the need to use different learning environments under certain circumstances, depending on the students' needs (McKenna, 2004; Gunbatar and Karalar, 2018). Personality traits reflect in students' patterns of thoughts, feelings, and behaviors like cognitive preferences

(Theodoropoulos et al., 2017). It seems that there are connections between personality traits and students' perceptions about problem solving abilities in more established fields like Maths (Grežo and Sarmány-Schuller, 2018). But even in CSE, a study by Theodoropoulos et al. (2017) found that cognitive personality dimensions provided significant results in correlation with playing games for acquiring basic programming knowledge. The present work aims to further investigate the aspect of personality traits and programming learning tasks, so the following question is researched.

RQ2: Is there any difference in students' cognitive styles and their preferences for creating games in different programming environments?

Rationale: Identify if students find some programming tools easier than others in terms of usability and learning, in correlation with their cognitive style.

There is also an increasing interest in research for another personality trait that is Emotional Intelligence (EQ) (Smith et al., 2014). Its importance is theoretically described (Ivanović et al., 2014) and researchers recognize that it is now time to incorporate this research in technology (Davis, 2020). It is also known that students' emotional states have an effect on their learning processes (Faria et al., 2010). For these reasons, the current work also focuses on EQ aspects with programming learning environments, wishing to move relevant research forward, since very little exists in possible connections between EQ and programming learning.

RQ3: Is there any difference in students' EQ and their preferences for creating games in different programming environments?

Rationale: Identify if students find some programming tools easier than others in terms of usability and learning, in correlation with their EQ.

Therefore, the current work focuses on children's personal learning characteristics, such as gender, cognitive style and EQ, through the design and creation of games in three different programming environments, wishing to move relevant research forward, since very little exists in possible connections between them (Theodoropoulos et al., 2020).

Our study aims to highlight gaps and future path for game design in programming learning environments in correlation with personal learning characteristics.

To answer the aforementioned questions, we employed a between-subjects design, with three experimental conditions. Different students tested each condition and formed three groups, so that each group is only exposed to a single user interface. Group A created games with Scratch, group B used App Inventor and made games for mobile devices, while group C created interactive stories-games with Alice 3D. All three, are part of the Greek school curriculum and are taught around eight teaching hours (each one). However, for the purposes of this study, we organized longer game-design workshops, in which students' groups spent around two months using the environments (eight weeks), to significantly increase their exposure to the environments and give them the opportunity to explore them in depth. In this study, we investigate: (i) the role of gender in different programming learning environments, and specific personality characteristics, that is (ii) the cognitive style and (iii) the emotional intelligence.

Finally, we research the longitudinal emotional effects from the game-design workshops, to discover relationships between students' perceptions that are not related to the above variables. More specifically we research:

RQ4: How do students' feel about the workshop' s life for creating games in different programming environments after a long time?

Rationale: Investigate the longitudinal effects of the programming environments in students' perceptions about, usability, learning experience, emotions regarding the workshop' s life.

This study is a natural next step to contribute to recent scholarly efforts use individual learning preferences to maximize learning effectiveness in programming education. The increasing exposure of children to programming as a compulsory part of their internationally creates a clear demand for scholars to evaluate tools and approaches, and offer frameworks to ensure education is delivered effectively in this curricular area (Theodoropoulos et al., 2020).

The rest of this paper is structured as follows. The related work section offers (i) a brief introduction to the programming environments used in this study, (ii) main aspects of the "learning programming by designing games" approach, and (iii) background work on programming preferences related with the variables studied in this work (gender and personality traits). The method section describes the procedure for conducting this study (context, participants, design, materials, data collection and data analysis). The results section presents the analysis for the selected RQs, while the discussion section discussed implications of this study (based on RQs) and provides guidelines for programming learning environments that respect individual learning preferences and aim to maximize learning effectiveness. Next, comes the limitations and future work section that discusses flaws or shortcoming of this work and lays out suggestions for future studies. Finally, the paper restates the study in the conclusion section.

RELATED WORK

Programming Environments of This Study

Previous research highlights the need to use different learning environments under certain circumstances, depending on the students' needs (Roth, 1999; Dillenbourg et al., 2002). Our study focuses on three programming learning environments widely known and used (1) Scratch (Maloney et al., 2010); (2) App Inventor (Wolber et al., 2015); and (3) Alice (Kelleher and Pausch, 2007). All three are part of the compulsory computing curriculum in Greek schools. Alice is considered a very good tool for programming learning. It has been used with children of different ages (Rodger et al., 2010) and abilities (Van Camp, 2013). It was found that it can help both younger and older kids to tackle programming issues (Cordova et al., 2011) and boost their interest in computer science (Distler, 2013). It can also help students with weak mathematical background to grasp computing concepts (Harrison, 2013) and to improve the interest of both genders in programming (Kelleher and Pausch, 2007). Alice is also useful for collaborative activities like pair programming, especially with children that are friends (Werner et al., 2013).

Similarly, MIT App Inventor is a visual programming environment for creating applications for Android-based smartphones and tablets (Perdikuri, 2014). Because of its simplicity and ease of use, it is considered an ideal environment toward computing democratization (Wolber et al., 2015) which could help underrepresented members of society to handle computing concepts. For example, it has helped young underrepresented children to work together and enhance their programming skills (Rahman, 2018). It works well with older and younger children, like primary school students (Alifah et al., 2019) and it allows hands-on experience even with complicated matters like controlling robot actions (Kline, 2018). Due to the App Inventor educational potential, it is important to train teachers (Dodero et al., 2017) and also support the teacher-student connectivity and communication (Ramos et al., 2015).

Finally, Scratch is a visual programming environment that allows children (mainly 8–16) to learn programming. Scratch also supports collaborative and self-directed learning through games and storytelling (Maloney et al., 2010). It is considered beneficial to start from a young age, like elementary school children (Hermans and Aivaloglou, 2017). In a simplified form, Scratch can also support children to create projects and not only solve puzzles, encouraging exploration and experimentation (Tsur and Rusk, 2018). With carefully planned activities, children seem capable of reflecting on their own learning and show increased motivation to continue with programming tasks (Dhariwal, 2018). Although certain concerns have been raised in the past regarding the potential of Scratch in allowing further involvement with programming beyond a visual language (Meerbaum-Salant et al., 2011), different studies showed that children that were first involved in programming through Scratch were perfectly capable of proceeding with professional textual languages like C#, Java (Armoni et al., 2015) and Python (Dorling and White, 2015).

Learning by Designing Games

According to Rieber et al. (1998), learning through designing games is more beneficial than traditional approaches. Game design considers that the act of creating a game is a road to learning in and of itself, independent of whether or not the game is enjoyed by others (Isbister et al., 2010). The concept of "learning by designing" is founded on the premise that active engagement in the design and development process is the most effective way to learn. This method is becoming more popular in programming education (Theodoropoulos and Lepouras, 2020).

Important factors in the learning process are the role of the teacher, the learning environment (in the conventional sense), and the instructional design. The learning environments discussed above, are akin to tools such as integrated development environments (IDE) and so forth, which are commonly understood in HCI/CS contexts as workspaces and software packages. However, we want to make clear the implicit distinction when there are blended (physical and digital) environments, as well as peer learning and collaborative learning pedagogies (Blackmore et al., 2011), which are things that take place in the learning environment of the classroom, learning management system, social space and so on. So, we used the three programming learning environments as game development environments within educational settings.

The potential of games to become a vehicle in programming education has been of interest in the research and education community (Theodoropoulos and Lepouras, 2020). Students are more active in the learning process through game-based learning and especially when they learn by developing games themselves (Kafai, 1995). However, more important than an individual tool or software package is a teacher's pedagogical content knowledge (PCK) (Park and Oliver, 2008), as this is what fosters the immediate learning environment for students (including opportunities for collaboration and peer learning) and makes knowledge accessible to students. Our study, necessarily and pragmatically focus on a discrete aspect of the learning process, as experimental approaches demand.

Student Preferences

The main variables and the conceptual model of this study are shown in **Figure 1**. The model is comprised of four main categories including: Gender, Cognitive Style, Emotional Intelligence and Effects from game-design workshops. As mentioned earlier, this research aims to examine students' individual learning preferences by creating games with different programming tools. Therefore, we chose to work with categories like gender and personality traits, which have been recognized in the literature as of high importance in programming education (e.g., McKenna, 2004; Smith et al., 2014; Theodoropoulos et al., 2017, 2020; Denner et al., 2019; Davis, 2020). However, these categories are characterized by diverse variables. Given that this work is one of the few so far regarding game-design through different programming environments in correlation with individual learning preferences, we selected the variables as shown in **Figure 1**.

Naturally, there are different user programming preferences (Chamberlain, 2017). Students have different preferences with regards to coding assignments. Important factors that influence choices seem to be task dependent such as task difficulty, potential of enhanced learning experience and perceived benefit (Smith et al., 2014). However, there are other factors that influence preferences which are task independent, like special learning demands. For instance, students with dyslexia need to be carefully considered when a programming learning task is designed (Powell et al., 2004).

Moreover, although Gal et al. (2017) found that creativity, as an important learning element in programming learning environments, is associated with contextual variables and not personal ones, there are studies that show the importance of personality in programming learning activities (Theodoropoulos et al., 2017). In fact, Theodoropoulos et al. (2017) used the Myers and Briggs Type Indicator (MBTI), a tool to identify cognitive preferences for receiving and processing information, to study possible differences in students' performance as they

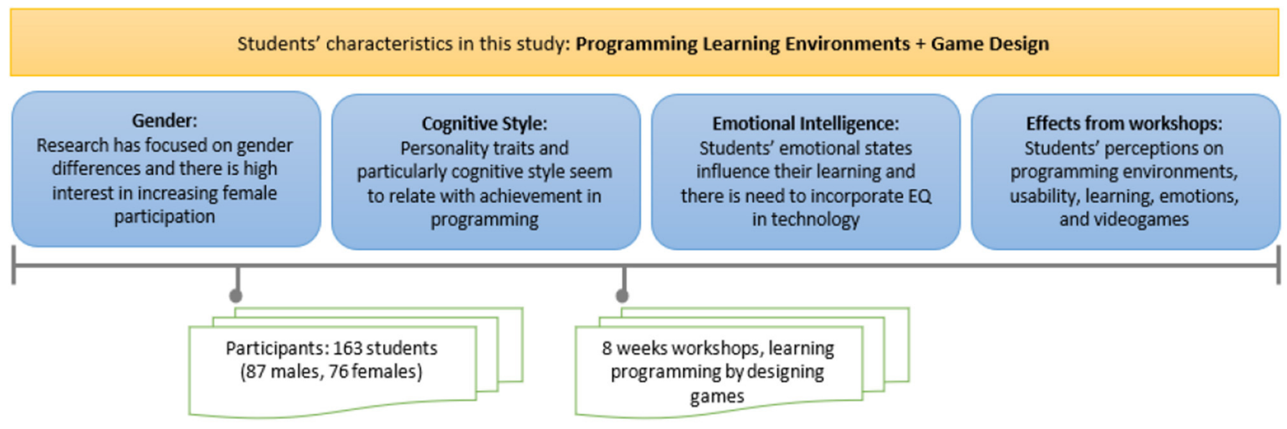
Students' characteristics in this study: Programming Learning Environments + Game Design
Gender: Research has focused on gender differences and there is high interest in increasing female participation
Cognitive Style: Personality traits and particularly cognitive style seem to relate with achievement in programming
Emotional Intelligence: Students' emotional states influence their learning and there is need to incorporate EQ in technology
Effects from workshops: Students' perceptions on programming environments, usability, learning, emotions, and videogames
Participants: 163 students (87 males, 76 females)
8 weeks workshops, learning programming by designing games

FIGURE 1 | Variables (conceptual model) used in this study.

engaged with programming learning activities. They found that 3 out of the 4 MBTI cognitive dimensions provided significant results. Introverted students and Judgers did significantly better than other personality types. In addition, students with high intuition also perceived the tasks as easier. The present work aims to further investigate the aspect of cognitive preferences and programming learning tasks, also using the MBTI tool. Studies using other cognitive style assessment tools, like the Big Five did not find any significant correlations (Alhathli et al., 2017). Personality has been successfully used to better understand user preferences (Tkalcic and Chen, 2015). As early as 1978, Lee and Shneiderman (1978) found associations between personality and programming preferences. In addition, Osman et al. (2012) reported how specific learning environments like Scratch show positive correlations in terms of attractiveness with preferences of school students with specific personalities (i.e., Perfectionism), predicting learning motivation.

Gender seems to be another important and well-recognized factor in programming learning. Although studies repeatedly report not finding any differences in the programming style between the genders (e.g., McKenna, 2004), there seem to be significantly less girls in programming classes in all educational levels, from primary school to high school (McBroom et al., 2020). However, although the quality and style of programming seems to be similar for both boys and girls there are other significant differences, like girls reporting more negative emotions in programming learning activities (Coto and Mora, 2019). Girls also spend more time communicating with others in computer learning environments and show less interest in programming activities (Bruckman et al., 2002). There are differences in software usage, in exploring software features and reported confidence (Burnett et al., 2010). Finally, collaborative programming seems to be better received by girls (Vandenberg et al., 2020). For these reasons, the current study will also consider gender issues.

Finally, over the last years there is an increasing interest in Emotional Intelligence (EQ) research. Its importance is theoretically described (Ivanović et al., 2014) and researchers recognize that it is now time to incorporate this research in technology (Davis, 2020). It is also known that students' emotional states have an effect on their learning processes (Faria et al., 2010). For these reasons, the current work also focuses on EQ issues and programming learning environments, wishing to move relevant research forward, since very little exists in possible connections between EQ and programming learning.

METHOD

Context and Participants

The participants of this study, were students attending a medium-sized high school in Greek secondary education (with around 180 students in total). Students from seven different school classes were chosen under the proposal of their teachers. They were informed about the study and its purpose from their CS teacher and all agreed to participate in it. Their parents signed for them to participate. Then they received a thorough explanation of the study and explained that no rewards will be given. So, in our game-design workshops took part 163 students (mean age = 14.6 years), 46.6% female and 53.4% male students. The game-design workshops held within their regular school program and curriculum. The study was conducted during the school year 2019–2020, by one of the researchers/authors of this paper with the help of the school's CS teacher. All workshops held in the school's IT laboratory. Moreover, we had in our mind that students may modify their behavior in response to their awareness of being observed (Hawthorne effect). Literature shows that little can be securely known about the conditions under which behaviors operate, their mechanisms of effects, or their magnitudes (McCambridge et al., 2014). So, the fact that this was a long study (eight weeks) and that everything was held in the normal school program with the participation of the CS teacher, ensures that this bias is reduced. Finally, it is expected that the year 2021–2022 all students will graduate the high school.

Design and Materials

In our experimental research (**Figure 2**), three students' groups where formed. Group A consisted of 57 students (29 males and 28 females) and used the Scratch programming environment

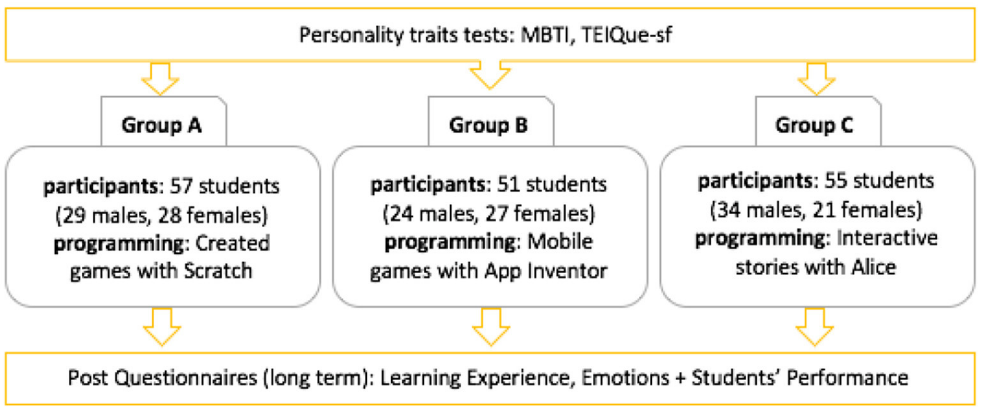
Personality traits tests: MBTI, TEIQue-sf
Group A
participants: 57 students (29 males, 28 females) programming: Created games with Scratch
Group B
participants: 51 students (24 males, 27 females) programming: Mobile games with App Inventor
Group C
participants: 55 students (34 males, 21 females) programming: Interactive stories with Alice
Post Questionnaires (long term): Learning Experience, Emotions + Students' Performance

FIGURE 2 | The design of this study.

TABLE 1 | The 8-week workshop process followed.

Week	Activities
Week 1	Present programming environments and experiment
Weeks 2–3	Experience the programming tools with curriculum worksheets and exercises
Weeks 4–7	Design and develop games with teacher guidance (following the process of **Table 2**)
Week 8	Share games by presenting them to their classmates

TABLE 2 | Game development process followed.

Stages	Description of game design activities
Discover	Explore game types/resources and see previous games made by others in each programming environment
Design	Design game characters and story, create or combine graphics to create game sprites and backdrops
Develop	Give life to the designs, program sprites/components/objects and behaviors
Debug	Playtest the game, fix bugs, add elements and improve the quality of gameplay
Deliver	Present game to classmates and get feedback

for creating games. Next, Group B with 51 students (24 males and 27 females), used App Inventor to create mobile games. Finally, the third group, Group C, consisted of 55 students (34 males and 21 females) and experienced Alice 3D, to create interactive stories-games.

All the programming environments are introductory in programming education and are designed to prevent programming errors common to beginners (e.g., syntax errors). Therefore, instead of typing commands they are based on blocks. All are part of the compulsory Greek secondary programming education (it depends on the instructor), in different age-groups. Scratch is used widely between K-7 to K-9 grades, while App Inventor and Alice are part of the K-10 grade. Every participating student in this study had some prior experience with Scratch.

This between-subjects research was conducted during the first semester of the 2019–2020 school year, before the pandemic of COVID-19 and lasted eight weeks, as shown on **Table 1**. As already mentioned, each group experienced a different programming environment in order to create games, however by the time we collected the post activity questionnaires (a year later), all students had experienced all the 3 programming tools.

Table 2 gives a summary of the five-step game design process followed at this study (students had to design games to learn programming).

The games had both designing and coding parts. Regarding the games' graphics students were able to use cc images form the web. They worked in groups, and they were free to work and create a game as they wanted. At the regular weekly workshops, students reported on any problems that they encountered, and they could receive immediate feedback from their peers or from the CS teacher.

Data Collection and Measures

There were four instruments used in the present study including, pre- and post-activity questionnaires.

Pre-questionnaire—Programming Experience

At the beginning of the workshops, participants filled a basic questionnaire including demographic questions and past exposure to programming. Then students completed two validated surveys to reveal their personality preferences, as follows:

Personality—Cognitive Style

We used the short version Myers and Briggs Type Indicator (MBTI) to identify basic personality aspects of the participants. With 20 questions, the tool allows the identification of cognitive preferences (how people prefer to receive and process information) in four dimensions:

TABLE 3 | Post-questionnaire Variables.

Item	Description	Coding	References
1	The programming environment was easy to use	EOU	Adams et al. (1992)
2	I feel confident that I acquired/learned basic programming skills?/concepts	COL	Theodoropoulos et al. (2017)
3	Which programming tool/environment do you like most?	LIK	Lee et al. (2009)
4	Which programming tool/environment is better for learning?	LEA	Lee et al. (2009)
5	What emotions do you have when you think back on the workshops experience? *HAP* for Happiness, *HOP* for Hope, *PRI* for Pride, *REL* for Relief, *ANG* for Anger, *DIS* for Discomfort, *ANX* for Anxiety and *SHA* for Shame	EMO	Frommel et al. (2018)
6	I play regular videogames	GAF	Theodoropoulos et al. (2017)

- Extraversion-Introversion (E-I), whether people focus more on others or on themselves,
- Sensing-Intuition (S-N), how people collect information,
- Thinking-Feeling (T-F), how people make decisions and
- Judging-Perceiving (J-P), how people organize their environment in order to make quick decisions.

MBTI is very popular in applied settings (Capraro and Capraro, 2002) and when compared to the construct of Big Five, there are strong correlations between its dimensions that support the validity and reliability of the tool (Furnham, 1996). In addition, MBTI has been successfully used in the past with children and has shown clear cognitive preferences in regards to programming (Theodoropoulos et al., 2017).

Personality—Emotional Intelligence

The Emotional Intelligence Questionnaire (Petrides et al., 2006) as a Personality Trait-short version (TEIQue-sf) is based on the detailed form of TEI-Que and consists of 30 closed-ended questions. Students were asked to choose an answer through a t-scale Likert. Then, the sum of the scores of their answers (maximum sum = 210, minimum sum = 30) are an indicator of their general EQ as a component of their personality. The TEIQue-sf questionnaire reveals four factors that evaluate the following characteristics:

- well-being,
- self-control,
- emotionality and
- sociability.

The four sub-factors can take values from 1 to 7, with higher values indicating well-being, self-control, emotionality, and sociability.

Post-questionnaire—Learning Experience and Emotions

Our analysis investigates the longitudinal effects of the programming environments (independent variables) in students' perceptions about, usability, learning experience, emotions regarding the workshop' s life (dependent variables). Emotions were coded in four positive items and four negative items. The questions used in the post activity questionnaire are presented in **Table 3**.

Students' Performance Data

For students' performance, we used the school's data in order to measure cognition. We adopted their CS course grade and their overall grade, from 2019–2020 school year.

Data Analysis

Being a between-subjects design and having different participants in each group, the current study fulfilled the independence of observations assumption. In addition, the independent variable consisted of three independent experimental groups and by using Likert scales we obtained ordinal data. All these factors lead to the use of One-Way ANOVAs for statistical analysis. Furthermore, and although the sample was adequate (more than 20 cases in each shell), we cannot assume normality. For this reason, together with One Way ANOVAs we also run non-parametric tests, i.e., Kruskal Wallis tests to increase the reliability of our results in case parametric validity conditions were not met. Except for the data provided by the questionnaires, our study gathered information from observations of the researcher and the teacher. This information provided a way to interpret and validate the results.

RESULTS

RQ1: Gender, programming environments and games.

To examine any possible gender differences in regard to the programming environments, an ANOVA test was performed. The results showed no significant difference between boys and girls, $F_{(1,47)} = 2.52$, $p = 0.28$. In gender preferences again, the One-way ANOVA test showed that there was a statistically significant difference in the boys from the Alice group, who found the programming tool difficult, $F_{(2,74)} = 7.369$, $p = 0.001$. Moreover, App Inventor and Alice seem to cause anxiety $F_{(2,74)} = 0.137$, $p = 0.051$ to boys while Alice made boys feel anger $F_{(2,74)} = 0.13$, $p = 0.063$. Girls found Scratch better for learning $F_{(2,65)} = 0.317$, $p = 0.028$, App Inventor second better and ALICE worst for learning. Males found Scratch easy to use and Alice second best.

RQ2: Cognitive style, programming environments and games.

Figure 3 presents the frequencies between the three groups and the students' cognitive style (MBTI).

For the cognitive styles, the Kruskal-Wallis H test showed that there was a statistically significant difference with extraverts who

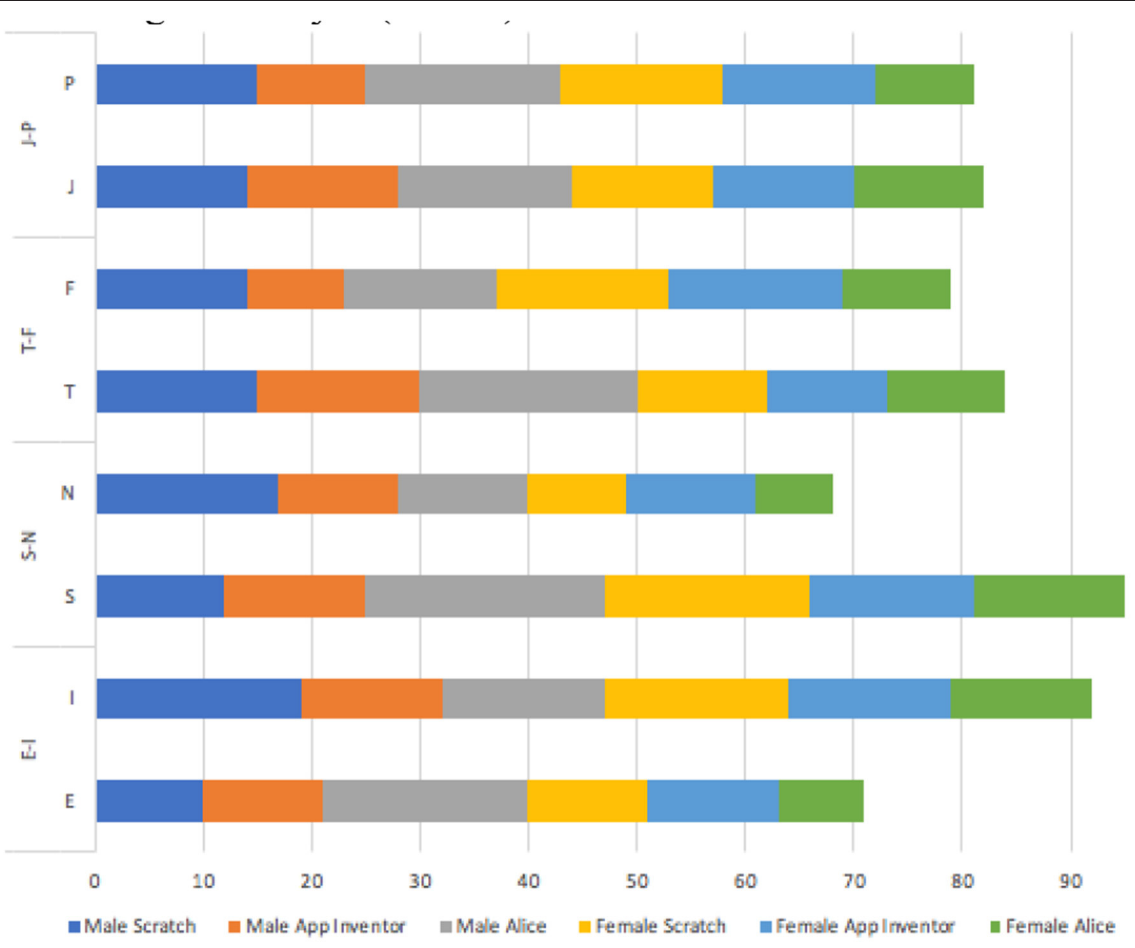

FIGURE 3 | Gender and cognitive styles (MBTI) per group.

found App Inventor easy to use $\chi^2(2) = 6.585$, $p = 0.014$, with a mean rank score of 49.00 for Scratch, 44.06 for App inventor and 36.69 for Alice. Introverts also reported low anxiety for App Inventor and Alice., $\chi^2(2) = 5.314$, $p = 0.035$, with a mean rank score of 47.32 for Scratch, 37.06 for App inventor and 24.29 for Alice. Lastly, introverts, thinkers, feelers and judgers found Scratch easy to use while sensors found Alice easy to use.

RQ3: EQ, programming environments and games.

Figure 4 presents the frequencies between the three groups and the students' EQ from **Table 4** it appears that the 4 factors of children's EQ are developed. The Kruskal-Wallis H test showed that students with low self-control found Scratch easy to use $\chi^2(2) = 6.545$, $p = 0.011$, with a mean rank score of 48.03 for Scratch, 24.02 for App inventor and 26.44 for Alice.

Similarly, students with low well-being found Scratch easy to use. Those with low emotionality and low sociability found Scratch easy to use.

Moreover, **Figure 5** presents students works from the three programming tools (screenshots from the games and part of the block-based coding).

RQ4: Long-term effects, programming environments and students' perceptions about, usability, learning experience, emotions.

General, most students find Scratch the easiest to use environment between the three (Mean = 4.19, *SD* = 0.86), while second comes App inventor. Moreover, they believe that they can learn/acquire basic programming skills from all the environments almost equally. On the contrast, in the question "which programming tool/environment do you like most" (LIK), the majority of 60.7% reported App Inventor, with Alice coming next with 24.1% and finally Scratch with 15.2%. Concerning the perception of learning between the three environments (LEA), App Inventor got the 51%, Scratch 35.2% and Alice the 13.8%.

Regarding their emotion (**Figure 6**), students did not report strong emotions, but most reported happy about bringing in mind the workshops period ($n = 102$), (see also **Table 5**).

Finally, regarding their performance, all three groups showed high similarity in their performance rate (Group A: $M_1 = 14.36$, $SD_1 = 4.44$; Group B: $M_2 = 13.48$, $SD_2 = 5.34$; Group C: $M_3 = 14.12$, $SD_3 = 4.26$).

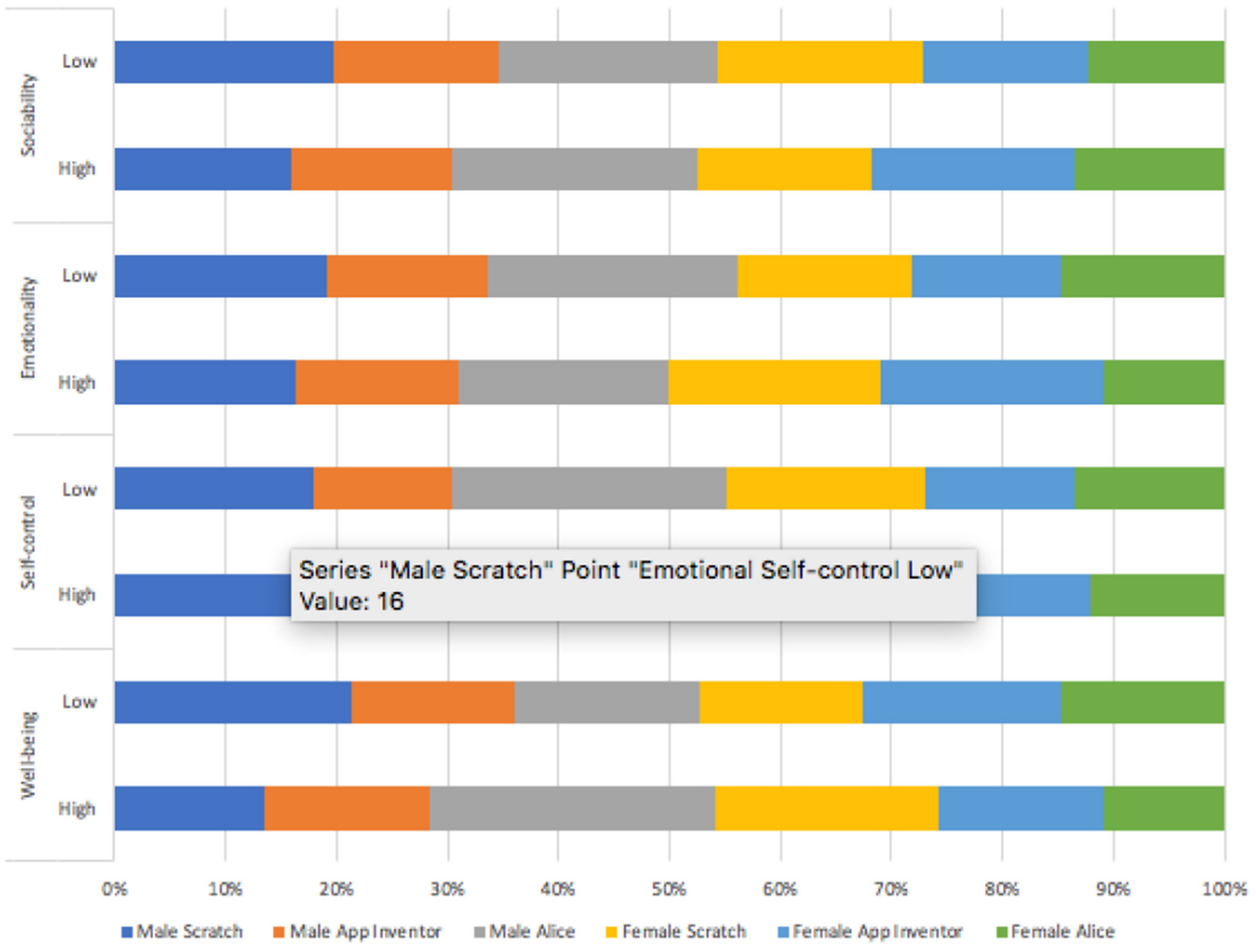

FIGURE 4 | Gender and emotional profiles (EQ) per group.

TABLE 4 | EQ profiles.

	Emotional							
	Well-being		**Self-control**		**Emotionality**		**Sociability**	
	High	Low	High	Low	High	Low	High	Low
	74	89	74	89	74	89	82	81
Std. Dev	1,01		0,94		0,92		1,05	
Mean	4,49		4,91		4,42		5,21	

DISCUSSION—GUIDELINES FOR DESIGNING GAMES IN SCHOOL

RQ1: Gender, programming environments and games.

In this study, most students seem to agree that Scratch is the easiest of the three programming learning environments for creating games. However, we found some different gender preferences regarding Scratch. Girls found it better for learning, whereas boys reported that it was easier to use. App Inventor seems to work well across genders. All students seemed to like it, since they could make games for their mobile phones. In addition, females also explicitly reported that App Inventor was a good learning tool. Girls, however, report that Alice is worse for learning, compared to the other two learning environments. This finding does not come in align with previous work (Kelleher et al., 2007), where Alice motivates girls to learn programming. However, in this study the tool was compared with the Alice without storytelling support. App Inventor seems to be widely

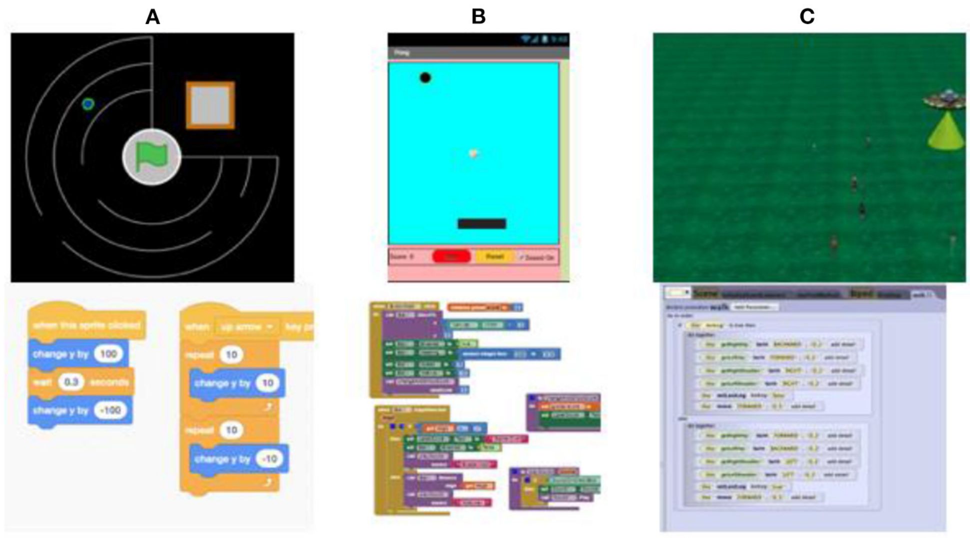

FIGURE 5 | Students works in **(A)** Scratch, **(B)** App Inventor and **(C)** Alice.

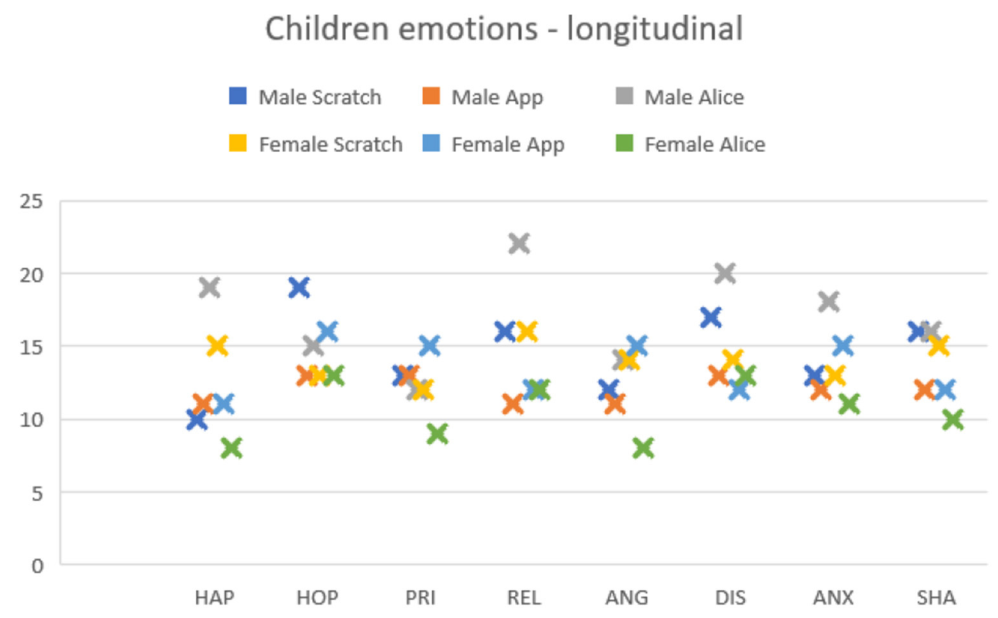

FIGURE 6 | Students' emotions about the gaming workshops after one school year.

accepted by students and we would recommend its use to instructors, after a programming introduction with Scratch, in align with (Papadakis et al., 2014).

RQ2: Cognitive style, programming environments and games.

We observed significant results with regards to participants cognitive style and Scratch' s usability. Introverts, Thinkers, Feelers and Judgers explicitly mentioned that Scratch was easy to use. Furthermore, most students found Alice difficult to use, apart from Sensors and Extraverts that also reported low anxiety levels (who found it easy to use). Therefore, with the exception of two cognitive styles, Extraverts and Sensors, Alice troubled students and we recommend caution in its use. Instructors should proceed with Alice keeping in mind that students might need more time, encouragement, and explanation. App Inventor seems to work well across cognitive styles. Only mild anxiety was reported by boys, although extraverts seem to report particularly low levels of anxiety with it and ease of use. Personality models and programming education need further research (Li et al., 2018), since as the present study shows, there are significant correlations that should be taken into account. Barroso et al. (2019) found

TABLE 5 | Summary of measurements.

Item	Constructs	Coding	Mean	SD
1	Ease of Use	EOU	4.19	0.868
2	Confidence for Learning	COL	4.26	0.808
3	Programming tool preference	LIK	2.09	0.623
4	Programming tool learning	LEA	1.79	0.669
5	Emotions	EMO-HAP	0.90	0.306
6		EMO-HOP	0.06	0.242
7		EMO-PRI	0.03	0.164
7		EMO-REL	0.02	0.143
8		EMO-ANG	0.05	0.214
10		EMO-DIS	−10.33	30.45
11		EMO-ANX	0.03	0.164
12		EMO-SHA	0.00	0.000
13	Game-playing frequency	GAF	4.01	0.920

that personality traits influence quality of software developed by students and that might be the case when creating games also.

RQ3: EQ, programming environments and games.

Similarly, participants with different EQ types seemed to prefer Scratch and finding it easy to use, like students with low self-control, low well-being, low emotionality, and low sociability. This is a very important finding since students with more vulnerable emotional characteristics find Scratch easier to use. Keeping this in mind, instructors could approach programming learning with Scratch to boost confidence in more emotionally vulnerable individuals. Thus, Scratch is recommended as a good starting environment for all students and once confidence levels are built, then instructors could proceed with other programming learning environments. App Inventor seems to work well across emotional styles. The effects of trait EQ on students' performance and related variables across education seem very important (Petrides et al., 2018) and also need further investigation.

RQ4: Long-term effects, programming environments and students' perceptions about, usability, learning experience, emotions.

Although students seemed to be happy with their programming experience and did not report strong negative emotions one year later, the use of Alice raises some concerns **Figure 6**. There are some gender preferences regarding the use of Alice and boys seem to report more negative emotions like anxiety and even anger. In any case, emotions are important in educational settings especially when dealing with technology and even more when creating software like games. Previous work shows that girls were happy when engaged with technology through creative development programming activities (Giannakos et al., 2014). Game-design is a creative process and has the potential to engage students, so carefully designed activities can have both motivating and learning results.

As a bottom line, in terms of general usage between the three programming tools of this work, Scratch seems to be the most appropriate for teaching younger students. App Inventor is possibly more appropriate for a more official introduction to programming and giving "tangible" results (e.g. games in students' mobile), while Alice may fit better when the aim is the transition into a conventional language.

Limitations and Future Work

Our work was conducted within formal learning settings with three different programming tools. Given the novelty of the study, we did not aim to measure participants' learning gains. However, this factor may limit the scope of a study that focuses on learning. For example, MBTI is not used to assess cognition as such. It is used to measure individuals' preferences and is not suitable for evaluating actual cognition. In a future follow up work, we will use a formal cognitive test to measure cognition and/or access students' games with a gaming rubric. So, we will expand this research design to align with the current curriculum learning aims and measure learning gains and efficacy.

Moreover, psychometric tools like MBTI have received criticism regarding their validity (Boyle, 1995), and caution is recommended for their use (Tzeng et al., 1984). Thus, the present work wishes to show that one teaching style and a single programming learning approach does not fit all. Additionally, although the behavior tools and personality tests that we used in our study are valid and reliable, maybe some students did not give appropriate responses, and this could affect the reliability of the results. However, the sample in our study counterpoises this possible issue. Moreover, a threat to the external validity of our experiment was the single use of school students with limited programming skills and experience. To mitigate this threat, we will proceed in our future work with older students with better programming skills and experience.

Finally, this study, necessarily and pragmatically focus on a discrete aspect of the learning process, as experimental approaches demand. Curriculum matters are briefly discussed as part of setting context. However there is limited critical reflection on the relationship between the wider curriculum and this study's intervention (the game-design workshops). Due to the nature of our workshops only students were recruited. In a future study we aim to develop a co-design workshop with K-12 schoolteachers to enhance both the learning and teaching effects in the educational settings.

CONCLUSION

The current work is among the very few that focus on personality aspects of school students and programming preferences through designing games. It reveals that different personalities have different preferences and learning needs and wishes to show that one teaching style and a single programming learning approach does not fit all. We want to raise awareness of personalization of instruction based on the specific cognitive and emotional styles. In this light, programming learning environments, for designing games and especially as a part of a curriculum to which a vast number of children is exposed, need to be better understood in terms of student preferences and needs. We also revealed the need for better instructor guidelines and clear steps when designing games in order to learn programming. Based on the perceived difficulty and complexity of use of the learning environments, we suggest that Scratch should be used first, then App Inventor and finally Alice. In addition, students with more vulnerable emotional profiles need careful handling and possibly better assistance when they are exposed to environments like Alice. Thus, the present work points out some student differences and the need to research further this area, in order to maximize learning benefits.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

ETHICS STATEMENT

Ethical review and approval was not required for the study on human participants in accordance with the local legislation and institutional requirements. Written informed consent to participate in this study was provided by the participants' legal guardian/next of kin.

AUTHOR CONTRIBUTIONS

Both authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

FUNDING

This work was supported by the National Contribution for University of the Peloponnese European Research Projects (years 2016–17) - CrossCult (Grant Number 84329).

ACKNOWLEDGMENTS

We are grateful to the students of the General High School of Tegea for participating in our study. We also thank our colleague Angeliki Antoniou, who helped in earlier phases of this work.

REFERENCES

Adams, D. A., Nelson, R. R., and Todd, P. A. (1992). Perceived usefulness, ease of use, and usage of information technology: a replication. *MIS Q.* 16, 227–247. doi: 10.2307/249577

Alhathli, M., Masthoff, J., and Siddharthan, A. (2017). Should learning material's selection be adapted to learning style and personality?," in *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization* (Bratislava), 275–280. doi: 10.1145/3099023.3099079

Alifah, P., Sudrajat, A., Sumantri, M. S., Satibi, O., and Utomo, E. (2019). "The Thematic learning module based on MIT APP Inventor 2 for 4th grade elementary school," in *Proceedings of the 2019 7th International Conference on Information and Education Technology* (Aizu-Wakamatsu: ACM), 170–173. doi: 10.1145/3323771.3323816

Armoni, M., Meerbaum-Salant, O., and Ben-Ari, M. (2015). From scratch to 'real' programming. *ACM Trans. Comput. Educ. (TOCE)* 14, 1–15. doi: 10.1145/2677087

Bancroft, P., and Roe, P. (2006). "Program annotations: feedback for students learning to program," in *Proceedings of the 8th Australasian Conference on Computing Education-Volume* 52 (Hobart, TAS), 19–23.

Barroso, A. S., Prado, K. H. de J., and Soares, M. S., and do Nascimento, R. P. (2019). "How personality traits influences quality of software developed by students," in *Proceedings of the XV Brazilian Symposium on Information Systems* (Aracaju), 1–8. doi: 10.1145/3330204.3330237

Ben-Ari, M. (1998). Constructivism in computer science education. *ACM Sigcse Bull.* 30, 257–261. doi: 10.1145/274790.274308

Blackmore, J., Bateman, D., O'Mara, J., Loughlin, J., and Aranda, G. (2011). *The Connections Between Learning Spaces and Learning Outcomes: People and Learning Places*. Geelong, VIC: Deakin University. Available online at: http://www.learningspaces.edu.au/docs/learningspaces-literaturereview.pdf (accessed July 10, 2014).

Bosse, Y., and Gerosa, M. A. (2017). Why is programming so difficult to learn? Patterns of difficulties related to programming learning mid-stage. *ACM SIGSOFT Softw. Eng. Notes* 41, 1–6. doi: 10.1145/3011286.3011301

Boyle, G. J. (1995). Myers-Briggs type indicator (MBTI): some psychometric limitations. *Aust. Psychol.* 30, 71–74. doi: 10.1111/j.1742-9544.1995.tb01750.x

Broll, B., Lédeczi, A., Volgyesi, P., Sallai, J., Maroti, M., Carrillo, A., et al. (2017). "A visual programming environment for learning distributed programming," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, WA), 81–86. doi: 10.1145/3017680.3017741

Bruckman, A., and Jensen, C., and DeBonte, A. (2002). *Gender and Programming Achievement in a CSCL Environment*.

Burnett, M., Fleming, S. D., Iqbal, S., Venolia, G., Rajaram, V., Farooq, U., et al. (2010). "Gender differences and programming environments: across programming populations," in *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Bolzano), 1–10. doi: 10.1145/1852786.1852824

Capraro, R. M., and Capraro, M. M. (2002). Myers-briggs type indicator score reliability across: studies a meta-analytic reliability generalization study. *Educ. Psychol. Measure.* 62, 590–602. doi: 10.1177/0013164402062004004

Chamberlain, R. D. (2017). "Assessing user preferences in programming language design," in *Proceedings of the 2017 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (Onward), 18–29. doi: 10.1145/3133850.3133851

Cordova, J., Eaton, V., and Taylor, K. (2011). Experiences in computer science wonderland: a success story with Alice. *J. Comput. Sci. Coll.* 26, 16–22. doi: 10.5555/1961574.1961577

Coto, M., and Mora, S. (2019). "Are there any gender differences in students' emotional reactions to programming learning activities?," in *Proceedings of the XX International Conference on Human Computer Interaction* (Donostia-San Sebastian), 1–8. doi: 10.1145/3335595.3335608

Davis, C. E. (2020). "Making time for emotional intelligence in production and technology," in *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks* (New York, NY), 1–2. doi: 10.1145/3388767.3407362

Denner, J., Campe, S., and Werner, L. (2019). Does computer game design and programming benefit children? A meta-synthesis of research. *ACM Trans Comput Educ (TOCE)* 19, 1–35. doi: 10.1145/3277565

Dhariwal, S. (2018). "Scratch memories: a visualization tool for children to celebrate and reflect on their creative trajectories," in *Proceedings of the 17th ACM Conference on Interaction Design and Children* (Trondheim), 449–455. doi: 10.1145/3202185.3202770

Dillenbourg, P., Schneider, D., and Synteta, P. (2002). "Virtual learning environments," in *Proceedings of the 3rd Hellenic Conference Information and Communication Technologies in Education* (Rhodes), 3–18.

Distler, C. (2013). "Piloting alice in the upper school," in *Proceedings of Alice Symposium on Alice Symposium* (Durham, NC), 1–5. doi: 10.1145/2532333.2532334

Dodero, J. M., Mota, J. M., and Ruiz-Rube, I. (2017). "Bringing computational thinking to teachers' training: a workshop review," in *Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality* (Cádiz: ACM), 1–6. doi: 10.1145/3144826.3145352

Dorling, M., and White, D. (2015). "Scratch: a way to logo and python," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (Kansas City, MO),191–196. doi: 10.1145/2676723.2677256

Faria, F. F., Veloso, A., Almeida, H. M., Valle, E., Torres, R, da S., and Gonçalves, M. A. (2010). Learning to rank for content-based image retrieval," *Proceedings of the international conference on Multimedia information retrieval* (Philadelphia, PA), 285–294. doi: 10.1145/1743384.1743434

Fessakis, G., and Prantsoudi, S. (2019). Computer science teachers' perceptions, beliefs and attitudes on computational thinking in Greece. *Inform. Educ.* 18, 227–258. doi: 10.15388/infedu.2019.11

Frommel, J., Schrader, C., and Weber, M. (2018). "Towards emotion-based adaptive games: emotion recognition via input and performance features," in *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play* (Melbourne, VIC), 173–185. doi: 10.1145/3242671.3242672

Furnham, A. (1996). The big five versus the big four: the relationship between the Myers-Briggs Type Indicator (MBTI) and NEO-PI five factor model of personality. *Pers. Individ. Diff.* 21, 303–307. doi: 10.1016/0191-8869(96)00033-5

Gal, L., Hershkovitz, A., Morán, A. E., Guenaga, M., and Garaizar, P. (2017). "Suggesting a log-based creativity measurement for online programming learning environment," in *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale* (Cambridge, MA), 273–277.

Giannakos, M. N., Jaccheri, L., and Leftheriotis, I. (2014). "Happy girls engaging with technology: assessing emotions and engagement related to programming activities," in *International Conference on Learning and Collaboration Technologies* (Cham), 398–409. doi: 10.1007/978-3-319-07482-5_38

Gomes, A., and Mendes, A. J. (2007). "An environment to improve programming education," in *Proceedings of the 2007 International Conference on Computer Systems and Technologies* (Bulgaria: ACM), 1–6. doi: 10.1145/1330598.1330691

Grežo, M., and Sarmány-Schuller, I. (2018). Do emotions matter? The relationship between math anxiety, trait anxiety, and problem solving ability. *Studia Psychol.* 60, 226–244. doi: 10.21909/sp.2018.04.764

Gunbatar, M. S., and Karalar, H. (2018). Gender differences in middle school students' attitudes and self-efficacy perceptions towards mBlock programming. *Eur. J. Educ. Res.* 7, 925–933. doi: 10.12973/eu-jer.7.4.925

Guo, P. J. (2018). "Non-native english speakers learning computer programming: barriers, desires, and design opportunities," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC), 1–14.

Harrison, J. (2013). "Alice in Virginia Beach, a continuing experiment," in *Proceedings of Alice Symposium on Alice Symposium* (Durham, NC: ACM), 1–6. doi: 10.1145/2581116.2532335

Hazzan, O., Gal-Ezer, J., and Blum, L. (2008). "A model for high school computer science education: The four key elements that make it!," in *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education* (Portland, OR) 281–285. doi: 10.1145/1352322.1352233

Hermans, F., and Aivaloglou, E. (2017). "To scratch or not to scratch? A controlled experiment comparing plugged first and unplugged first programming lessons," in *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (Nijmegen), 49–56. doi: 10.1145/3137065.3137072

Isbister, K., Flanagan, M., and Hash, C. (2010). "Designing games for learning: insights from conversations with designers," in *Proceedings of the Sigchi Conference on Human Factors in Computing Systems* (Atlanta, GA), 2041–2044. doi: 10.1145/1753326.1753637

Ivanović, M., Radovanović, M., Budimac, Z., Mitrovi,ć D., Kurbalija, V., Dai, D., et al. (2014). "Emotional intelligence and agents: survey and possible applications," in *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS*14) (Thessaloniki), 1–7. doi: 10.1145/2611040.2611100

Kafai, Y. (1995). *Making Game Artifacts To Facilitate Rich and Meaningful Learning.* New York, NY: Routledge.

Kafai, Y. B. (2012). *Minds in Play: Computer Game Design as a Context for Children's Learning. London: Routledge.*

Kelleher, C., and Pausch, R. (2007). Using storytelling to motivate programming. *Commun. ACM* 50, 58–64. doi: 10.1145/1272516.1272540

Kelleher, C., Pausch, R., and Kiesler, S. (2007). "Storytelling alice motivates middle school girls to learn computer programming," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, CA), 1455–1464. doi: 10.1145/1240624.1240844

Kiridoshi, Y., Ishibashi, K., and Kozono, K., Limura, I. (2018). "Initial consideration on designing a system to support science communication and continuous programming learning," in *Proceedings of the 10th International Conference on Education Technology and Computers* (Tokyo), 223–230. doi: 10.1145/3290511.3290557

Kline, R. (2018). An android smartphone as IDE and robot controller. *J. Comput. Sci. Coll.* 33, 137–138. doi: 10.5555/3205191.3205207

Lee, B. C., Yoon, J-, O., and Lee, I. (2009). Learners' acceptance of e-learning in South Korea: theories and results. *Comput. Educ.* 53, 1320–1329. doi: 10.1016/j.compedu.2009.06.014

Lee, J. M., and Shneiderman, B. (1978). "Personality and programming: time-sharing vs. batch preference," in *Proceedings of the 1978 Annual Conference-Volume 2*, 561–569.

Li, X., Shih, P. C., and Daniel, Y. (2018). "Effects of intuition and sensing in programming performance using MBTI personality model," in *Proceedings of the 2nd International Conference on Advances in Image Processing* (Chengdu), 189–193.

Makris, D., Euaggelopoulos, K., Chorianopoulos, K., and Giannakos, M. N. (2013). "Could you help me to change the variables? Comparing instruction to encouragement for teaching programming," in *Proceedings of the 8th Workshop in Primary and Secondary Computing Education* (Aarhus), 79–82. doi: 10.1145/2532748.2532761

Malik, S. I., and Coldwell-Neilson, J. (2018). Gender differences in an introductory programming course: New teaching approach, students' learning outcomes, and perceptions. *Educ. Inf. Technol.* 23, 2453–2475. doi: 10.1007/s10639-018-9725-3

Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E. (2010). The scratch programming language and environment. *ACM Trans. Comput. Educ. (TOCE)* 10, 1–15. doi: 10.1145/1868358.1868363

Mathew, R., Malik, S. I., and Tawafak, R. M. (2019). Teaching problem solving skills using an educational game in a computer programming course. *Inform. Educ.* 18, 359–373. doi: 10.15388/infedu.2019.17

Matthews, R., Hin, H. S., and Choo, K. A. (2009). Multimedia learning object to build cognitive understanding in learning introductory programming," in *Proceedings of the 7th international Conference on Advances in Mobile Computing and* Multimedia (Kuala Lumpur), 396–400. doi: 10.1145/1821748.1821824

McBroom, J., Koprinska, I., and Yacef, K. (2020). "Understanding gender differences to improve equity in computer programming education," in *Proceedings of the Twenty-Second Australasian Computing Education Conference* (Melbourne, VIC), 85–194. doi: 10.1145/3373165.3373186

McCambridge, J., Witton, J., and Elbourne, D. R. (2014). Systematic review of the Hawthorne effect: new concepts are needed to study research participation effects. *J. Clin. Epidemiol.* 67, 267–277. doi: 10.1016/j.jclinepi.2013.08.015

McKenna, P. (2004). Gender and black boxes in the programming curriculum. *J. Educ. Resour. Comput. (JERIC)* 4, 6. doi: 10.1145/1060071.1060077

Meerbaum-Salant, O., Armoni, M., and Ben-Ari, M. (2011). "Habits of programming in scratch," in *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (Darmstadt), 168–172. doi: 10.1145/1999747.1999796

Nagashima, K., Cho, S., Horikoshi, M., Manabe, H., Kanemune, S., and Namiki, M. (2018). "Design and development of bit arrow: a web-based programming learning environment," in *Proceedings of the 10th International Conference on Education Technology and Computers* (Tokyo), 85–91. doi: 10.1145/3290511.3290525

Ndaiga, W., and Salim, A. (2015). "Kids hacker camps in Kenya: hardware hacking effectiveness in skills transfer," in *Proceedings of the Seventh International Conference on Information and Communication Technologies and Development* (Singapore). doi: 10.1145/2737856.2737873

Osman, M. A., Loke, S. P., Zakaria, M. N., and Downe, A. G. (2012). "Secondary students' perfectionism and their response to different programming learning tools," in *2012 IEEE Colloquium on Humanities, Science and Engineering (CHUSER)* (Kota Kinabalu), 584–588. doi: 10.1109/CHUSER.2012.6504380

Papadakis, S., Kalogiannakis, M., Orfanakis, V., and Zaranis, N. (2014). "Novice programming environments. Scratch and app inventor: a first comparison," in *Proceedings of the 2014 Workshop on Interaction Design in Educational Environments* (Albacete), 1–7. doi: 10.1145/2643604.2643613

Papavlasopoulou, S., Sharma, K., Giannakos, M., and Jaccheri, L. (2017). "Using eye-tracking to unveil differences between kids and teens in coding activities," in *Proceedings of the 2017 Conference on Interaction Design and Children* (Standford, CA), 171–181. doi: 10.1145/3078072.3079740

Park, S., and Oliver, J. S. (2008). Revisiting the conceptualisation of pedagogical content knowledge (PCK): PCK as a conceptual tool to understand teachers as professionals. *Res. Sci. Educ.* 38, 261–284. doi: 10.1007/s11165-007-9049-6

Paul, J. L., Kouril, M., and Berman, K. A. (2006). "A template library to facilitate teaching message passing parallel computing," in *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (Houston, TX), 464–468. doi: 10.1145/1124706.1121487

Perdikuri, K. (2014). "Students' experiences from the use of MIT App Inventor in classroom," in *Proceedings of the 18th Panhellenic Conference on Informatics (*Athens), 1–6. doi: 10.1145/2645791.2645835

Petrides, K. V., Sanchez-Ruiz, M-. J., Siegling, A. B., Saklofske, D. H., and Mavroveli, S. (2018). Emotional intelligence as personality: Measurement and role of trait emotional intelligence in educational contexts," in *Emotional Intelligence in Education*, eds K. Keefer, J. Parker, D. Saklofske (Cham: Springer), 49–81. doi: 10.1007/978-3-319-90633-1_3

Petrides, K. V., Sangareau, Y., Furnham, A., and Frederickson, N. (2006). Trait emotional intelligence and children's peer relations at school. *Soc. Dev.* 15, 537–547. doi: 10.1111/j.1467-9507.2006.00355.x

Powell, N., Moore, D., Gray, J., Finlay, J., and Reaney, J. (2004). Dyslexia and learning computer programming. *ACM SIGCSE Bull.* 36, 242–242. doi: 10.1145/1026487.1008072

Qian, Y., Yan, P., and Zhou, M. (2019). "Using data to understand difficulties of learning to program: a study with Chinese middle school students," in *Proceedings of the ACM conference on Global Computing Education* (Chengdu), 185–191. doi: 10.1145/3300115.3309521

Rahman, F. (2018). "Leveraging visual programming language and collaborative learning to broaden participation in computer science," in *Proceedings of the 19th Annual SIG Conference on Information Technology Education* (Fort Lauderdale, FL), 172–177. doi: 10.1145/3241815.3242586

Ramos, D., de Moura Oliveira, P. B., and Pires, E. S. (2015). "APP inventor as a tool to reach students," in *Proceedings of the 3rd International Conference on Technological Ecosystems for Enhancing Multiculturality* (Porto Portugal: ACM), 311–316. doi: 10.1145/2808580.2808627

Rieber, L. P., Smith, L., and Noah, D. (1998). The value of serious play. *Educ. Technol.* 38, 29–37.

Rodger, S. H., Bashford, M., Dyck, L., Hayes, J., Liang, L., Nelson, D., et al. (2010). Enhancing K-12 education with alice programming adventures," in *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education* (New York, NY), 234–238. doi: 10.1145/1822090.1822156

Roth, W. M. (1999). Learning environments research, lifeworld analysis, and solidarity in practice. *Learn. Environ. Res.* 2, 225–247. doi: 10.1023/A:1009953920993

Rugelj, J., and Lapina, M. (2019). "Game design based learning of programming," in *Proceedings of SLET-2019–International Scientific Conference Innovative Approaches to the Application of Digital Technologies in Education and Research* (Stavropol–Dombay).

Ryokai, K., Lee, M. J., and Breitbart, J. M. (2009). "Multimodal programming environment for kids: a "thought bubble" interface for the Pleo robotic character," in *CHI'09 Extended Abstracts on Human Factors in Computing Systems* (Boston, MA), 4483–4488. doi: 10.1145/1520340.1520687

Schwartz, J., Stagner, J., and Morrison, W. (2006). "Kid's programming language (KPL)," in *ACM SIGGRAPH 2006 Educators Program* (Boston, MA). doi: 10.1145/1179295.1179348

Shanahan, J. (2009). "Students create game-based online learning environment that teaches Java programing," in *Proceedings of the 47th Annual Southeast Regional Conference* (Clemson, SC), 1–4. doi: 10.1145/1566445.1566545

Smith, T., and Gokhale, S., and McCartney, R. (2014). "Understanding students' preferences of software engineering projects," in *Proceedings of the 2014 Conference on Innovation and Technology in Computer Science Education* (Uppsala), 135–140. doi: 10.1145/2591708.2591753

Spieler, B., and Slany, W. (2018). Game development-based learning experience: gender differences in game design. arxiv [preprint].arxiv:1805.04457.

Theodoropoulos, A., Antoniou, A., and Lepouras, G. (2017). How do different cognitive styles affect learning programming? Insights from a game-based approach in Greek schools. *ACM Trans. Comput. Educ. (TOCE)* 17, 3. doi: 10.1145/2940330

Theodoropoulos, A., and Lepouras, G. (2020). *Digital Game-Based Learning and Computational Thinking in P-12 Education: A Systematic Literature Review on Playing Games for Learning Programming. Handbook of Research on Tools for Teaching Computational Thinking in P-12 Education*. Philadelphia, PA: IGI Global. doi: 10.4018/978-1-7998-4576-8.ch007

Theodoropoulos, A., Poulopoulos, V., and Lepouras, G. (2020). "Towards a framework for adaptive gameplay in serious games that teach programming: association between computational thinking and cognitive style," in *International Conference on Interactive Collaborative Learning* (Cham), 530–541. doi: 10.1007/978-3-030-68198-2_49

Tholander, J., Fernaeus, Y., and Holmberg, J. (2004). "Tangible programming and role play program execution for kids," in *Proceedings of the 6th International Conference on Learning Sciences* (Santa Monica, CA), 641–641.

Thomas, L., Ratcliffe, M., Woodbury, J., and Jarman, E. (2002). Learning styles and performance in the introductory programming sequence. *ACM SIGCSE Bull.* 34, 33–37. doi: 10.1145/563517.563352

Tkalcic, M., and Chen, L. (2015). Personality and recommender systems," in *Recommender Systems Handbook*, eds F. Ricci, L. Rokach, and B. Shapira (Boston, MA: Springer), 715–739. doi: 10.1007/978-1-4899-7637-6_21

Truong, N., Bancroft, P., and Roe, P. (2003). "A web based environment for learning to program," in *Proceedings of the 26th Australasian Computer Science Conference-Volume* (Heidelberg), 16, 255–264.

Tsur, M., and Rusk, N. (2018). "Scratch microworlds: designing project-based introductions to coding," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, MD), 894–899. doi: 10.1145/3159450.3159559

Tzeng, O. C., Outcalt, D., Boyer, S. L., Ware, R., and Landis, D. (1984). Item validity of the Myers-Briggs type indicator. *J. Pers. Assess.* 48, 255–256. doi: 10.1207/s15327752jpa4803_4

Van Camp, R. (2013). "Alice summer camps: evaluating multiple formats," in *Proceedings of Alice Symposium on Alice Symposium* (Durham, NC), 1–1. doi: 10.1145/2581116.2532345

Vandenberg, J., Tsan, J., Hinckle, M., Lynch, C., Boyer, K. E., and Wiebe, E. (2020). "Gender differences in upper elementary students' regulation of learning while pair programming," in *Proceedings of the 2020 ACM Conference on International Computing Education Research* (New Zealand: ACM), 311–311. doi: 10.1145/3372782.3408117

Von Hausswolff, K. (2017). "Hands-on in computer programming education," in *Proceedings of the 2017 ACM Conference on International Computing Education Research* (Tacoma, WA: ACM), 279–280. doi: 10.1145/3105726.3105735

Weintrop, D., and Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Trans. Comput. Educ. (TOCE)* 18, 1–25. doi: 10.1145/3089799

Werner, L., Denner, J., Campe, S., Ortiz, E., DeLay, D., Hartl, A. C., et al. (2013). Pair programming for middle school students: does friendship influence academic outcomes?," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, CO), 421–426. doi: 10.1145/2445196.2445322

Wolber, D., Abelson, H., and Friedman, M. (2015). Democratizing computing with app inventor. *GetMobile* 18, 53–58. doi: 10.1145/2721914.2721935

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.