Students’ Perceptions on Using a Dual Modality Programming Environment

Georgia Menounou, Stavroula Pantelopoulou, Margarita Karaliopoulou and Evangelos Kanidis

Abstract—In this paper we investigate students’ outcomes regarding the use of a dual modality programming environment that combines both text mode and block mode in class. The survey targets students of the 3rd grade of Junior High School (Gymnasium) in Greece and attempts to answer research questions regarding their perceptions after a one-year course using the programming environment Pencil Code. The analysis of the collected data clearly indicates that the vast majority of students have used both modes during their work. The students find the block mode easier than the text mode but prefer to modify or correct the code using the text mode. We also noticed significant correlations between the mode used and the students’ gender, their school grades and their future course choice. The findings of this study can be useful when redesigning new Informatics curricula for the secondary education.

Index Terms—Computer Science Education, Dual Modality Programming, Junior High School (Gymnasium), Pencil Code.

I. INTRODUCTION

Researchers constantly try to find effective ways in order to teach young students programming skills. Computer Science (CS) has dominated our lives and the future seems to be dependent on programmers’ competence. The core problem is not to produce young developers but to lay the foundation for developing an algorithmic and computational way of thinking. Programming promotes causal explanation and helps students to cope easily with time-varying phenomena, segment big problems to smaller ones, strengthen logic and improve their abilities to use, organize, plan and integrate information spread out in various environments. Effective teachers are motivated by the same question that has driven Andrea diSessa: “can education be transformed by the computer so that children can learn more, learn more easily at an earlier age, and learn with pleasure and commitment?” [1]. Furthermore, CS teachers are aware of the benefits of acquiring algorithmic thinking at early school stages.

Mannila et al. [2] back in 2006 assured us that a simple language not only can be used in introductory CS courses to help students acquire basic programming skills, but may provide basic skills to future professionals. As well there is no listed disadvantage in learning to program with a simple language and switch later on to a more complex one [2].

The Curricula of most countries in Europe have incorporated Informatics. In Greece it was first introduced in Junior High Schools in 1993 and henceforth Information and Communications Technologies (ICT) is taught for one hour a week in all grades. Since September 2016 all grades in every primary school have been teaching ICT for an hour a week. In the last two grades pupils are taught introductory programming courses, but it depends on the teacher to choose the programming environment for his/her class. In Gymnasium (junior high school) programming exists systematically -covering more than two thirds of the course material- only in the third grade where fundamental algorithms and programming structures are formally introduced, using Logo-like environments [3]. Since the school year 2017-18 simple programming structures have also been taught in the first and second grade of Gymnasium.

Programming environments for these ages are grouped in Block-based programming environments (Scratch, BYOB, Snap, AppInventor, Alice etc.), Text-based programming environments (MicroWords Pro, Python, Java etc.) and Hybrid-based programming environments -that translate from blocks to text- (Alice3, Pencil Code, Snap!, Patch, Tiled Grace, Caligo Jigsaw, Greenfoot’s Frame-based Editor, Scratch GP, DrawBridge etc.) [4]. Later in High School block-based environments are different per use (mobile app development with MIT App Inventor and Pocket Code; modelling and simulation tools including StarLogo TNG, DeltaTick, NetTango, and EvoBuild; creative and artistic tools like Turtle Art and PicoBlocks; and game-based learning environments like RoboBuilder, Lightbot, and Google’s Made with Code initiative) [5].

Text based environments (text-based editing/Command User Interface) are these in which the user has to type program commands. Usually, students are annoyed at the fact that they should learn the programming commands and type them correctly. Block-based environments use blocks of code, which are collated in order to create a program. Students can create with less effort more complex programs since they do not have to learn the syntax of a programming language. They are able to incorporate multimedia context such as art, music and interactivity to create games, stories and applications. Blocks are dragged into a scripting area and snapped together to form scripts [5]. Karaliopoulou and Kanidis [6] have recently published an overview of the bibliography, which discusses the advantages and disadvantages of block-based programming and includes a comparison of it with text-based programming.

Recently, block-based and text-based environments are
supported by some dual modality environments (hybrid block/text environments) that blend features of both modalities. We believe that Dual-Modality environments are better in teaching young students the fundamentals of programming. We chose Pencil Code among others, and maintain that it will help students who struggle to connect block-based programming to ‘real programming’. Furthermore, Dual-Modality environments help them move to text-based environments since they have already been in touch with code, have memorized commands, have seen a debugger’s messages, know what a syntax error is, etc. Pencil Code is a dual modality programming language in which users can switch between tiled and textual view. Their program is editable in both forms. The environment supports editing large subsets of CoffeeScript as blocks and text and encapsulates an animated transition between two modes. Switching between block and textual view is possible at any time, except when textual syntax errors exist [7], [8], [9], [10].

Dual modality environments as Pencil Code try to bridge the gap by letting the student decide whether and when he/she is ready for the transition to the text-based mode. Notwithstanding, as Weintrop and Wilensky [11] mention, little is known about the relationship between modality and novice programmers’ emerging practices.

In Greece block-based programming is dominant in primary schools and the first two grades of gymnasium, and the problem to be solved is how to make the transition to text-based programming easier. Karaliopoulou, Apostolakis and Kanidis [3] investigated teachers’ perceptions regarding the use of block and text programming environments in the class. According to their research the majority of teachers believe that introducing students to block-based programming environments first eases the transition to text-based programming later. Their work tried to determine the appropriate age of students for transition from block-based programming to text-based programming. Teachers believe that students in Primary school and the first two grades of the Gymnasium should use a block-based environment, and should be introduced to text-based programming in the 3rd grade of the Junior High School, when students are 14 years old.

Karaliopoulou and Kanidis [6] proposed for third grade of Gymnasium parallel working in a block-based environment and a text-based environment; they used code.org and MicroWords Pro. Their small scale research indicated that blocked-based programming ‘looks like a game’ but text-based programming makes students feel like ‘professionals’. Later on their proposal [12] was implemented with more students, and a quantitative survey with a sample of 166 students took place. Their research was conducted among students, who first had to solve a problem using block-based programming and then solve the same problem in a text-based environment, using a method called “Mediated Transfer of Knowledge”. They concluded that most students thought that their prior experience with block-based programming helped them in creating the program in the text-based environment.

Considering the afore-mentioned results, we carried out a research in Greece based on a course where we used the Pencil Code environment for one hour a week in the 3rd grade through the whole school year in order to facilitate the transition from block to text programming. We found that Pencil Code has many positive effects for teachers and students: it is online and does not require installation, no email account is required, the files that are produced per each account are stored on the internet, students are not coerced to have a specific position in the lab, students may login from their homes and study using the material that is produced in the classroom, they are capable of using material that can be browsed through other users’ accounts and they share their work.

In this paper we investigate the opinion of the students based on their classroom experience with Pencil Code. We first provide the theoretical background related to our research questions. Then we state the research questions and present our research results. Finally, we discuss our results and point out the most interesting findings.

II. BACKGROUND

Block-based programming has succeeded in introducing programming to novices. “Jigsaw puzzle” metaphor makes users feel comfortable from the beginning. Nevertheless, students who move from block-based environments to text-based ones believe that they are transferred to a new and totally different environment. Learners do not consider visual languages “real” programming languages. This view is supported by preconceived ideas about what is programming, the fact that visual languages are easier and that textual programming is later found to be difficult. Some users even find textual languages and syntax intimidating. Educators try to facilitate this transition [9], [10], [13].

There are numerous efforts to support this transfer, mostly with the aim to use the computational thinking skills which students have gained through their involvement with the block-based environment in order to support their development to text-based programming. One example is Robinson who introduced Patch -a modified version of Scratch 1.4- that will ease the transition from Scratch to Python [14]. Moreover, there are researches that try to facilitate the interface of block-based programming languages (e.g. grouping blocks by functionality) in order to improve usability [15].

Price and Barnes analyzed the outcome of a study in two groups of novice programmers -one assigned to block interface and the other to text interface- and came to the conclusion that “students using the block interface spent less time off task and completed more of the activity’s goals in less time”. The chosen environment was Tiled Grace that allows the user to switch between the two modes. In their article there is an extended description of block-based environments along with references to previous evaluations of such software. There are reviews of studies that compare textual and block programming environments and reviews of studies about the transition from block-based to textual programming [16].

Weintrop and Wilensky [5], provide two cases of systematic comparisons of block-based and text-based environments. In a study using Snap!, they came to the conclusion that students perform “better on block-based questions related to conditional logic, function calls and
definite loops” and found “no differences on questions related to variables, indefinite loops, and program comprehension questions”. The second study they mention is the one of Price and Barnes [17] in which authors found little difference in learning outcomes though they report that students working in block-based environments complete activities at a faster rate.

Aiming at bridging the gap between text-based and block-based programming tools, dual mode environments have been developed, allowing students to interfere with both blocks and text. Some of them enable students to code in text and transform it to block and vice-versa: one of them is Pencil Code. “The goal of Pencil Code is to build enough confidence in beginning programmers so that they can create programs without Pencil Code”. They are able to have error free results very quickly by toggling between text-based and block-based environments. Students write “real” CoffeeScript using blocks. Bau et al. [18] used it in classrooms ranging from grade 6 to college. A test took place in eight public schools in rural South Carolina where students were given four after school lessons using the software. None of them had previous programming experience. Authors recorded whether students had been editing their programs using texts or blocks. It was proved that “students shifted from programming in blocks almost all the time, to working in text about 95% of the time” and the most valuable observation is that students “were never required to program in text” [18]. Bau “believes that text languages will always be a staple of computer science education” [19].

Bau, Gray, Kelleher, Sheldon, & Turbak [20] declare that block-based programming “has become lingua franca” for introductory course of coding. The basic advantages are listed as follows: novice programmers can easily recognize the structures and memorize them while cognitive load reduces and errors are being prevented. Professionals are not left behind since new frameworks are able of accessing a variety of APIs, enabling diversity and experimentation. This very interesting paper includes a section called “Measuring Learning Transfer” which accumulates cases that support the hypothesis of effective learning transfer from block languages to traditional textual languages. A number of studies, in K12, Universities, Colleges and separate CS courses indicate that learning with blocks and using mediated transfer pedagogy improve students’ engagement and student’s grades along with the percentage of them that takes further CS courses. The paper mentions some state of the art programming alternatives that enable novice programmers to build programs even for domains such as 3D printing or Querying the Semantic Web [20]. Pencil Code is subject to researchers’ addition in order to improve tool’s debugging process and users’ understanding [21].

In a small scale research in Japan, comparing the programming education for primary schoolchildren with VPLs (Visual Programming Languages) and that with TPLs (Textual Programming Languages), the VPL approach proved to be more suitable for increasing the motivation of the learners [22].

The new tendency is to study the use of block-based and hybrid-based environments and the way they support transferring student’s programming skills to text-based ones. This produces the following research questions: 1. Does hybrid-based allow students to understand programming concepts (Variables, Looping, Procedures, and Conditional Logic) easier when they turn to text-based programming? 2. Will the hybrid-based environment reduce the shock of transferring to a text-based environment?

Weintrop [23] used Pencil Code to compare the programming skills that have been acquired by three groups of students who used the environment differently. The first group used only blocks, the second only text and the third one used hybrid-based environment. All students followed the same curriculum for 5 weeks and afterwards they moved to java and text-based programming. His findings were very interesting: 92% of students consider block-based programming is easier to learn and includes less commands to memorize, but on the other hand is difficult to build a project complex enough [23].

Weintrop and Wilensky [24] report “a proliferation of introductory programming environments” for the past ten years. Scratch, Alice, and other block-based programs are popular for novice programmers. Recently, block-based and text-based environments have stopped being separate. Until now, most teachers have been trying to assist the transition between these two modalities. The answer from researchers is the design of a dual modality environment that supports both text-based and block-based programming and hybrid block/text environments that blend features of both modalities. Authors make an extended review of tools that support the transition from block-based to text-based environments and others that provide bi-directional support - Pencil Code is one of those. Pencil code’s modification that is introduced here (pencil.cc) is a hybrid approach that has blocks that can be browsed and used by drag-and-drop, while text editor supports authenticity and increases programming speed. One can see all of three isomorphic interfaces [24].

Blanchard [25] assumes that if we prove that hybrid environments bridge the gap between block-based and text-based, they may serve as scaffolding for moving from basic CS topics to more advanced ones. Author tries to figure out if learning environments impact students’ attitudes regarding CS and if their confidence for their ability depends on it.

Brown, Kölling, & Altadmri [26] describe how advantages of block-based systems will be available to proficient programmers by adding keyboard support. This is a sound evidence for the usefulness of hybrid environments. Some of the limitations of block-based programming that can be overcome with keyboard support are the speed of entry and manipulation, and the support of code navigation (jumping between usage and definition of an entity). Authors have developed a frame editor as an addition to a block-based environment and by using CogTool -a software system that automates the analysis of interaction sessions (e.g. key presses, mouse clicks, eye movement, reading time)- they found that the prototype frame-based editor is the fastest in insertion, modification, deletion, movement, replacement compared to Scratch, Alice and NetBeans.

Kölling, Brown, & Altadmri [27] in a detailed paper published by ACM propose frame-based editing in order to
bridge the gap between block-based programming and text-based programming. The transition between these two systems is an educational challenge. Plain text, according to the authors, is the “the norm for proficient and professional programmers and the accepted educational goal for programming in many countries”. The UK is a working example of the above situation. Computing National Curriculum for England states that children aged 7-11 should be capable of using “sequence, selection and repetition in programs; work with variables and various forms of input and output”. This need is met by block-based programming, mostly with Scratch. 11-14 year-olds must “use two or more programming languages, at least one of which is textual, to solve a variety of computational problems”. Python, Java, Visual Basic are used for the switch to text-based programming. At the age of 12 students face this transition. Researchers developed a frame-based editor that supports a new language called Stride, intergraded in the Greenfoot development environment, which is a Java-like language that causes significantly less frustration caused by unnecessary syntax errors.

Homer and Noble [9] conducted an experiment using Tiled Grace and their results indicated that users find dual views helpful. They found that participants generally (76%) enjoyed using the environment; that enjoyment was lower for more experienced users, suggesting that these interfaces are suitable for novice programmers. According to Homer and Noble [9] Visual programming environments (Scratch, Blockly, Alice) “present a limited language” and programmers must move on a “real” textual programming. Researchers underline that in many cases programmers feel like they have to learn to program over again. Three years later, Homer and Noble [10] published their results. They found both positive and cautionary notes for block-based programming in the results. Two of the questions of their survey were: a. which tasks users preferred to perform as text and which as tiles and b. how these patterns vary by the experience level of the user. They ran a usability experiment trialling Tiled Grace with 33 participants and they focused on usability and engagement. They had the opportunity to examine which modality -blocks or text- is preferred for different tasks. One of their findings was that participants used the other view to work around limitations of the one they were using. “Users in an unfamiliar language prefer to use the blocks”, “more-experienced users were substantially less favorable towards a block-based environment than less experienced users”, “block languages for programming education must have an exit strategy”, “drag-and-drop visual editing is a significant problem for some users” and “providing multiple views of code helps users be more comfortable with it”. They identified the need for extra research to confirm or refute their findings.

III. RESEARCH QUESTIONS

Based on the theoretical background above, a questionnaire was devised and provided to students who were in the 3rd grade of junior high school. The questionnaire was based on the following research questions:

1. What is the percentage of students that had an introduction to programming and what type of programming environment had they used before Pencil Code?

2. According to students, which mode of Pencil Code (block or text) is closer to a “real” programming language?

3. Do the students switch between text-based and block-based mode of Pencil Code during the activities?

4. Which mode of Pencil Code do students find easier
   a) For writing a program?
   b) For finding coding errors?
   c) For bypassing difficulties?

5. Which mode of Pencil Code do students prefer to work with
   a) When they write a program?
   b) When they make changes in code?

6. Which mode of Pencil Code do students prefer to work with at the beginning of the school year and which at the end of the school year?

7. What type of programming environment would students prefer for next year’s course if a course in programming exists?

8. Are there any correlation between:
   - Demographic elements (gender, future courses choice, math grade, grades at the first semester)
   - Students’ perception of which mode of Pencil Code is closer to a “real” programming language
   - Their choice of switching between text-based and block-based mode
   - Their perception of which mode of Pencil Code is easier to work with (when they write a program, for finding coding errors, for bypassing difficulties)
   - The mode they prefer to work with (when they write a program during the activities, at the beginning of the school year, at the end of the school year and when making changes)
   - The programming environment they would choose at next year’s course.

IV. MATERIALS AND METHODS

At the beginning of the school year 2017-2018, IT teachers were invited to participate in a seminar about Pencil Code programming environment. There were 26 teachers in this seminar, and 15 out of them participated in our research. Teachers involved in the research, taught a one-year course in programming, using the Pencil Code environment, at 3rd grade junior high school students. All teachers gave students the same activities, prepared in advanced and tested in one school by two coordinators. Students had to use Coffeescript commands and the python turtle to create different activities. Activities were simple at first and got more advanced by the end of the school year. The teachers participated in meetings where the students’ progress was discussed and their work was coordinated. At the end of the school year a questionnaire was given to their students. Therefore, the survey’s reference population was 3rd grade junior high school students (688 individuals) in Greece. The sampling was purposive [28] and it was part of a big area in the north of Athens and East Attica where 124 junior high schools exist.
Quantitative research methods were used to explore the attitudes, values and beliefs of the research population [29]. The tool used to conduct the study was a structured and anonymous questionnaire addressing the research issues associated with this study. The construction of the questionnaire was driven by the intention of providing quantitative results that could be analyzed. During the investigation, the research team respected the anonymity and confidentiality of the respondents [30].

The questionnaire consisted of two parts. The first part included demographic questions whereas the second part included questions concerning factors related to the preferred mode of Pencil Code programming environment (blocks vs. text). The questionnaire, in Greek, was created in Google forms and the responses were received anonymously. Overall, 35 variables were coded. The statistical package SPSS (Statistical Package for Social Sciences v.24) was used to perform statistical analysis. The association between different variables was tested using the $\chi^2$ independence test. For significance level $a=0.05$, a result was considered statistically significant when $p\text{-value}<0.05$ [31]. Before distribution, the questionnaire was submitted to a content and face validity test. It was reviewed by Informatics teachers at different schools and experts in the field to obtain feedback from an outside party who is less invested in the instrument. In order to ensure its consequence and the stability of its results internal consistency reliability technique was applied for group of variables; the Cronbach's alpha coefficient was $a=0.73$.

V. RESULTS

A. Participants Profile

The research took place during the school year 2016-17. The sample consisted of 688 students, 329 boys (47.8%) and 357 girls (52.2%), who attended the 3rd grade of Junior High school (students are aged 14-15). “Informatics” lesson is compulsory in the Greek high school curriculum. 32.7% of students declare that they wish to continue with Theoretical Studies in Literature and Art and 67.3% with Mathematics and Science. 94.3% of students owns a computer, but only 24.1% of them performed programming at home. 67.6% of students had been assigned a general grade from 16 to 20, at the end of the first semester, according to a scale of 0-20. Students prefer to collaborate while coding (76.5%).

B. Data Description

Students were asked if they had worked with programming environments other than Pencil Code in the previous years. These results are summarized in Table I.

<table>
<thead>
<tr>
<th>TABLE I. PROGRAMMING ENVIRONMENT PREVIOUSLY USED IN CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
</tr>
<tr>
<td>5th Grade Primary School</td>
</tr>
<tr>
<td>6th Grade Primary School</td>
</tr>
<tr>
<td>1st Grade High School</td>
</tr>
<tr>
<td>2st Grade High School</td>
</tr>
<tr>
<td>3rd Grade High School</td>
</tr>
</tbody>
</table>

19.6% of students had no previous experience in programming before 3rd Grade. It is worth mentioning that students having no experience in programming before 3rd Grade and chose to work with blocks at the beginning of the course were 13.81% of all students. This percentage is close to those that had no experience in programming before 3rd Grade and chose to work with blocks at the end of school year (12.06% of all students). On the contrary, those who had previous experience in programming and chose to work with blocks at the beginning of the course were 56.10% of all students but this percentage decreased: only 40.41% preferred blocks at the end of the school year. (See fig1, fig2).

When students were asked to express their opinion about whether text-based or block-based programming mode is closer to a “real programming language”, 67.4% of students replied text-based and 32.6% block-based.

Participants believe that it is easier to code in block-based mode (67.2%) than in text-based one (32.8 %). Only 38.8% of them choose coding in text, and more than half of them (52.3%) prefer to modify programs with the text mode. Moreover, 48.3% prefer to bypass difficulties using the text mode of Pencil Code. By the end of the school year the percentage of students that preferred to use the text mode of Pencil Code increased from 30.1% to 47.5% (see Table II).
It is very important that 447 students (65%) switched between the two modes in order to complete their programming tasks.

During the school year coding tasks became more difficult and complicated. As tasks were getting more demanding, students preferred to use the text mode of Pencil Code. 60.3% of them answered that they chose the block-based mode for designing number with simple draw commands, 56.5%, for designing a square using a loop command and 47.2% for creating a function.

Only half of them (49.3%) will choose programming as next year’s course. If a programming course is obligatory then 45.8% will choose a text based environment and 56.1% want to continue with Pencil Code. At the end of school year 67% of students said they liked programming with Pencil Code.

C. Statistical Analysis

In order to determine whether variables are associated, we examined if several correlations are worth mentioning.

Student’s opinion regarding of what is closer to a programming language, text or blocks, was associated with grades in the first semester ($\chi^2=21.410$, df=4, $p=0.00<0.05$), with math score ($\chi^2=21.476$, df=4, $p=0.00<0.05$), with future courses choice (that is studying Mathematics and Science or Theoretical Studies in Literature and Art) ($\chi^2=10.567$, df=1, $p=0.01<0.05$), with their choice of working between text or blocks ($\chi^2=13.406$, df=1, $p=0.00<0.05$), with their choice to switch between text or blocks ($\chi^2=42.383$, df=1, $p=0.00<0.05$) and easier for finding coding errors ($\chi^2=15.493$, df=1, $p=0.00<0.05$), with their choice of mode for making changes ($\chi^2=56.661$, df=1, $p=0.00<0.05$) and at the end of school year ($\chi^2=67.597$, df=1, $p=0.00<0.05$), with the mode they believe that is easier to use for bypassing difficulties in coding ($\chi^2=20.922$, df=1, $p=0.00<0.05$), with their choice to switch between text and blocks ($\chi^2=9.992$, df=1, $p=0.02<0.05$) and with their choice between text coding or blocks for the next year’s course ($\chi^2=76.495$, df=1, $p=0.00<0.05$) It is worth mentioning that students with higher grades tend to prefer to work in text based mode (See fig3).

Similarly, student’s choice during lessons to work with the text or block based mode is associated with gender ($\chi^2=4.353$, df=1, $p=0.037<0.05$) (See fig4), with future courses choice ($\chi^2=4.220$, df=1, $p=0.04<0.05$, Phi=-0.07, $p=0.037$), with their perception of which type of programming mode was easier in general ($\chi^2=10.088$, df=1, $p=0.00<0.05$) and easier for finding any mistakes in code ($\chi^2=15.312$, df=1, $p=0.00<0.05$), with the mode they choose, when they made changes ($\chi^2=86.252$, df=1, $p=0.00<0.05$), at the beginning of school year ($\chi^2=90.168$, df=1, $p=0.00<0.05$) and at the end ($\chi^2=161.410$, df=1, $p=0.00<0.05$), with the mode they believe that is easier to bypass difficulties in coding ($\chi^2=54.510$, df=1, $p=0.00<0.05$) and with their choice between text coding or blocks as a next year course ($\chi^2=177.124$, df=1, $p=0.00<0.05$).

It is remarkable that 65.5% of those who choose text mode for exercises find this mode easier for finding coding mistakes and 50.4% of those who chose blocks find text mode easier than blocks (see Table III).

TABLE II. VARIABLES’ FREQUENCY TABLE

<table>
<thead>
<tr>
<th></th>
<th>text based mode</th>
<th>block based mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>226</td>
<td>462</td>
</tr>
<tr>
<td>Percent Frequency</td>
<td>32.8</td>
<td>46.2</td>
</tr>
<tr>
<td>Percent</td>
<td>67.2</td>
<td>53.8</td>
</tr>
</tbody>
</table>

| Choice between Text or Block for finding errors | 387 | 301 | 43.8 |
| Choice between Text or Block for changes in code | 360 | 328 | 47.7 |
| Text or Block mode was easier for bypassing difficulties | 332 | 356 | 51.7 |
| Choice between Text or Block at the beginning of school year | 207 | 481 | 69.9 |
| Choice between Text or Block at the end of school year | 327 | 361 | 52.5 |

Similarly, student’s choice during lessons to work with the text or block based mode is associated with gender ($\chi^2=4.353$, df=1, $p=0.037<0.05$) (See fig4), with future courses choice ($\chi^2=4.220$, df=1, $p=0.04<0.05$, Phi=-0.07, $p=0.037$), with their perception of which type of programming mode was easier in general ($\chi^2=10.088$, df=1, $p=0.00<0.05$) and easier for finding any mistakes in code ($\chi^2=15.312$, df=1, $p=0.00<0.05$), with the mode they choose, when they made changes ($\chi^2=86.252$, df=1, $p=0.00<0.05$), at the beginning of school year ($\chi^2=90.168$, df=1, $p=0.00<0.05$) and at the end ($\chi^2=161.410$, df=1, $p=0.00<0.05$), with the mode they believe that is easier to bypass difficulties in coding ($\chi^2=54.510$, df=1, $p=0.00<0.05$) and with their choice between text coding or blocks as a next year course ($\chi^2=177.124$, df=1, $p=0.00<0.05$).

It’s interesting that 71.9 % of those who choose text mode wish to continue studying Mathematics and Science and 28.1% Theoretical Studies in Literature and Art.

DOI: http://dx.doi.org/10.24018/ejers.2019.0.CIE.1292
We also noticed significant correlation between student’s perception of which type of programming mode is easier in general, text or block, and the mode they choose when they make changes (χ²=23.493, df=1, p=0.01<0.05), the mode they choose to work at the beginning of school year (χ²=18.138, df=1, p=0.01<0.05), the mode they choose to work at the end (χ²=32.531, df=1, p=0.00<0.05), the mode they choose to switch between text and blocks (χ²=4.774, df=1, p=0.01<0.05), their choice between text coding or blocks as a next year course (χ²=10.307, df=1, p=0.01<0.05).

There is a statistically significant association between student’s opinions regarding the type of programming mode that is easier for finding mistakes in code and the mode they choose, when they make changes (χ²=23.493, df=1, p=0.01<0.05), the mode they choose to work at the beginning of school year (χ²=18.348, df=1, p=0.00<0.05), the mode they choose to work at the end (χ²=32.531, df=1, p=0.00<0.05), the mode they believe that is easier to bypass difficulties in coding (χ²=64.576, df=1, p=0.00<0.05), their choice to switch between text and blocks (χ²=4.774, df=1, p=0.01<0.05), their choice between text coding or blocks as a next year course (χ²=10.307, df=1, p=0.01<0.05).

We also noticed significant correlation between the mode they choose, when they make changes with the future courses choice (χ²=10.308, df=1, p=0.01<0.05), with students’ math score (χ²=12.690, df=4, p=0.01<0.05), with the mode they choose at the beginning of school year (χ²=13.027, df=1, p=0.00<0.05) with the mode they choose at the end of school year (χ²=92.421, df=1, p=0.00<0.05), with the mode they believe that is easier to bypass difficulties in coding (χ²=47.841, df=1, p=0.00<0.05), and with their choice to switch between text and blocks (χ²=5.841, df=1, p=0.01<0.05).

The students choice of mode at the beginning of school year is significantly correlated with gender (χ²=10.011, df=1, p=0.02<0.05), with grades in the first semester (χ²=10.896, df=4, p=0.028<0.05), with math score (χ²=13.291, df=4, p=0.010<0.05), with the mode they prefer at the end of school year (χ²=21.129, df=1, p=0.00<0.05), with the mode they believe that is easier to bypass difficulties in coding (χ²=14.780, df=1, p=0.00<0.05), with their choice to switch between text and blocks (χ²=5.525, df=1, p=0.01<0.05) and with their choice between text coding or blocks as a next year course (χ²=34.541, df=1, p=0.00<0.05).

Correlation was also found between the mode student choose at the end of school year and future courses choice (χ²=4.434, df=1, p=0.035<0.05), the mode they believe that is easier to use for bypassing difficulties in coding (χ²=21.293, df=1, p=0.00<0.05), their choice between text coding or blocks as a previous course (χ²=187.168, df=1, p=0.00<0.05).

The mode they believe that is easier for bypassing difficulties in coding is associated with their choice between text coding or blocks as a next year course (χ²=30.382, df=1, p=0.00<0.05). Students’ choice to switch from text to blocks is associated with future courses choice (χ²=5.044, df=1, p=0.025<0.05) and with grades in the first semester (χ²=13.791, df=4, p=0.008<0.05). Finally, students’ choice between text and block based mode for next year course is associated with gender (χ²=15.101, df=1, p=0.00<0.05) and future courses choice (χ²=11.751, df=1, p=0.001<0.05).

There is a significant difference between the mode students preferred to work with at the beginning (M=1.70, SD=0.459) and at the end of school year (m=1.52, SD=0.500, t(687)=7.421, p=0.00<0.05). 60.9% of students who preferred text at the beginning still preferred text at the end and 39.1% preferred blocks. At the end 47.5% preferred text and 53.5% blocks (See fig5).

DOI: http://dx.doi.org/10.24018/ejers.2019.0.CIE.1292
that students prefer to work in a block-based programming environment but correct the code in text-based mode. In accordance with previous findings by Bau et al. [8], this study indicates that the majority of students (65%) switched between the two modes in order to complete their programming tasks.

By the end of the school year the percentage of students that prefer to use the text mode of Pencil Code from 32.8% reached the 47.5%. 61% of those that choose blocks at the beginning, at the end prefer to work in text mode. These findings indicate that students prefer to use the text mode more than the block mode after a period of use.

We found statistical significant correlations between many of the variables that we study. It is interesting that as students grade increase students prefer to code with text mode rather than the block mode of Pencil Code. We also found that boys prefer to use the text mode of Pencil Code more than the girls. Furthermore, the majority of students who choose the text mode wish to study Mathematics and Science in the future. Finally, students with experience in programming prefer to use the text mode of Pencil Code at the end of school year.

We believe that our results could be useful to stakeholders when designing new curriculum in Informatics, especially concerning the proposed type of programming environments per age.

VII. FUTURE WORK

The authors’ future plans are to study a dual mode programming environment that incorporates Python. Also, we plan to study the same variables more explicitly with a more representative sample of students throughout Greece.

ACKNOWLEDGMENT

We would like to thank all our students who participated in the survey for this research project and our colleagues who used Pencil Code experimentally in their classrooms.

REFERENCES

[31] I. Apostolakis and A. Stamouli, “Validity and reliability assessment of quantitative research questionnaires in health units: The case of a...


**Georgia Menounou** was born in Peloponnissos Greece. She graduated with the bachelor degree on Information Science and Technology from the Athens University of Economics and Business in 1993. She received a MED Degree from the Hellenic Open University in 2015.

She worked as a system and network administrator in the private sector and as a project manager in many EU and company projects. She also worked for several years as a teacher of Informatics in secondary education. She is currently responsible for supporting schools and teachers to implement ICT (Information Computer Technology) in teaching. Her research focuses mainly on the educational aspects of informatics.

Ms. Menounou is a member of PEKAP (PanHellenic Association of Teachers in Informatics) and of PAPEDE (PanHellenic Pedagogical Society for Secondary Education).

**Stavroula Pantelopoulou** was born in Kalamata, Greece. She obtained a degree in Informatics from the University of Piraeus in 1998, an MSc in Information Technology Management from the University of Sunderland and an MSc in Educational Technology and Human Resources Management from the University of Athens.

She worked as a developer, analyst and project manager in various retail, public sector and banking projects, and as an R&D analyst in the banking sector. She was a tutor in Piraeus University of Applied Science. She is currently a teacher of Informatics in secondary education. She published more than 10 articles in various scientific journals and international conference proceedings. She is a Reviewer of Emerald’s journal “The Learning Organization”.

Ms. Pantelopoulou is a member of PEKAP (PanHellenic Association of Teachers in Informatics) and the Economic Chamber of Greece.

**Margarita Karaliopoulou** was born in Athens. She obtained a degree in Mathematics from University of Patras in 1991, an MSc in Informatics and Operational Research in 1993, an MSc in Statistics and Operational Research in 2004 and a PhD from University of Athens with a dissertation entitled “Discrete Semi-Markov Procedures and applications in word Occurrences”.

She has worked for several years as a teacher of Informatics in secondary education. Since November 2017, she has been an EDIP member in the Department of Mathematics at the National and Kapodistrian University of Athens. Author of a book and several papers among which the paper entitled "On Discrete Time Semi-Markov Chains and Applications in Words Occurrences" Communications in Statistics-Theory and Methods. 37(8) 1306-1322 (Chrysaphinou O., Karaliopoulou M and Limnios N.).

Dr. Karaliopoulou is a member of the Hellenic Mathematical Society and of PAPEDE (Panhellenic Association of Secondary education) and cofounder of PEKAP (Panhellenic Association of teachers in Informatics).

**E. Kanidis** was born in North Greece. He studied Mathematics at the University of Athens. He received the MSc in Informatics from the Informatics Department of Athens University and the PhD in Cognitive Science from the Interdisciplinary Graduate Program of Athens University in "Basic and Applied Cognitive Science", in 2007.

He worked as a teacher in secondary education for several years and as a scientific researcher and tutor in Piraeus University of Applied Science. He is currently an educational consultant in Informatics at the Ministry of Education. His research focuses mainly on the educational aspects of informatics and on ways to improve classroom IT teaching in primary and secondary education. He authored two books, co-authored nine books and published more than 40 articles in various scientific journals and international conference proceedings.

Dr. Kanidis is a member of the Greek Computer Society (EPY), co-founder of PEKAP (Panhellenic Association of Teachers in Informatics) and of PAPEDE (Panhellenic Association of Secondary Education).