Exploring Approaches to Teaching Programming in 12th Grades Students. A Study after 20 Years of Teaching

Spyridon Doukakis
Department of Informatics, Ionian University, Corfu, Greece
Email: sdoukakis@ionio.gr

***Abstract*—From the school year 1999-2000, the course entitled "Informatics" is taught to 12th grade students who attend the scientific field "Sciences of Economy and Informatics" in Greek secondary education. The school students of this field wish to participate in the national examinations to be admitted to university departments of economics and informatics. The general purpose of the course is for school students to develop analytical and synthetic thinking, to acquire methodological skills and to be able to solve problems in a programming environment. The purpose of the study is to explore educators' views on teaching algorithm in the 12th grade in order to explore whether interventions and changes in the way of approaching teaching are required. For this purpose, 1127 computer science educators who teach the course "Informatics" in schools participated in a survey to explore these issues. The results of the present study focus on the ways in which educators approach the basic algorithmic structures (sequence, selection and loops), data structures and modular programming as well as the ways in which the computer lab is used for teaching the content of the course. According to the results of the research, it seems that one out of three educators wants to approach the course with a specific programming language instead of a hypothetical Greek programming language designed specifically for the course. In addition, 50% of the educators gives great emphasis on the virtual execution of programs. At the same time, six out of ten use the computer lab in some way during the course, for the teaching needs of the course. The results show a diversity in educators' teaching approaches when teaching algorithmic structures of sequence, selection and loops. Particularly high diversity is observed during the teaching of modular programming. Finally, the results of the research show the importance of educators' training in issues concerning their pedagogical content knowledge and their technological content knowledge.**

***Index Terms*—Algorithmic approaches, programming, quantitative research, teaching informatics**

I. Introduction

Relatively recently, courses in secondary education that attempt to help students develop their algorithmic and computational thinking have included. Thus, the teaching of algorithms and programming in secondary education is one of the goals of many educational systems worldwide [1], [2]. Several studies are related to how algorithmic concepts are introduced and their teaching approach in secondary education [2], [3].

In Greece and after a reform of the educational system that took place the period 1997-1999, a course in algorithmic teaching and programming was included for the first time in 12th grade of secondary education. The course was entitled "Application Development in a Programming Environment" and is taught from the school year 1999-2000. The school year 2019-2020 was renamed "Informatics". The course is taught especially to 12th grade students who attend the scientific field "Sciences of Economy and Informatics" and wish to participate in national exams to be admitted to university departments of economics and informatics.

The course has been taught for over 20 years in secondary education and has significantly influenced the way in which programming is introduced in higher education [4]. Both the educators' teaching experience and the research that has been carried out all these years, have been contributed to issues of algorithmic teaching and programming. In this article, an evaluation of the course is attempted, through the findings of existing research. In the next sections, the course and the relevant research are presented, in order to highlight good teaching practices and approaches for planning relevant courses in other educational contexts.

II. The Course

TABLE I. Syllabus of the Course "Informatics"

Sections	Content
Analysis of Problem	
Introduction to Programming	
Basic Programming Concepts	Data Types, Operators Constants, Variables Commands, Input-output Sequence Selection Loops
Data Structures	Arrays Stack Queue Searching Sorting

Manuscript received October 25, 2021; revised January 23, 2022.

doi: 10.18178/ijlt.8.1.15-22

Modular programming	Subprograms Procedures and functions Parameters
Program Debugging	

Despite the title of the course, which is called "Informatics", its content focuses on solving mainly computational problems and on algorithmic design of their solution (see Table I). According to the syllabus of the course, its general purpose is for school students to develop analytical and synthetic thinking, to acquire methodological skills and to be able to solve problems in a programming environment [5].

In the initial design of the course and for the first five years the students had the opportunity to solve the problems algorithmically, that is, to create the algorithm that solves the problem, either using pseudocode, or using a hypothetical Greek programming language called LANGUAGE designed specifically for the course, or in a programming language such as BASIC and Pascal.

After the first three years, the possibility of demonstrating problem solving with a programming language was removed and the use of pseudocode and LANGUAGE was strictly defined, while for the last five years, students have the opportunity to develop algorithms exclusively with the hypothetical Greek programming language. designed specifically for the course (LANGUAGE).

The course consists of a teaching package that includes the student's book, the student's book with exercises and the teacher's book [5]. In addition, it is suggested that the course must be taught in the computer lab. Although students at national level answer questions and create algorithms on paper, the educator has the opportunity to go with his/her students to a laboratory environment (in the computer lab of the school unit), so that students can work in a suitable programming environment [6], [7]. For this purpose, certified software has been developed that can enriches the laboratory characteristics of the course.

The course was a completely new subject in school education. Given that it has been taught in schools for more than twenty years, exploring the views, practices, attitudes and perceptions of the educators who teach the course has contributed to the discussion concerning a) the necessity of an algorithmic lesson in secondary education and b) the benefits that students can derive from attending a relevant course. The result of the scientific discussions highlighted the value of an algorithmic course in school education and the importance it has for high school graduates. In this context, the next section provides a literature review of algorithmic teaching and programming in secondary education. Next, the results of the research concerning a) the teaching approaches of the algorithmic structures (sequence, selection and loops), data structures and modular programming and b) the computer labs use are presented.

III. Literature Review

The inclusion of the course "Informatics" in the curriculum of the 12th grade specifically for students who wished to study in higher education in university departments of informatics and economics, was a pioneering event for secondary education in Greece. At that time, other countries opened the relevant discussion, so that to include programming courses in their curricula [8].

The course design attempted to emphasize the algorithmic approach and the development of problem-solving skills within a programming environment, so that students do not focus on cultivating and developing programming techniques and learning a specific programming language. However, in the following years, both due to the international context in which students in secondary education usually worked with a particular programming language and due to the fact that educators had learned programming as university students with real programming languages instead with a hypothetical Greek programming language designed specifically for the course, issues related to the value of this approach had been arisen [9], [10]. As a result of this discussion, the development of algorithms with the hypothetical Greek programming language designed specifically for the course became mandatory from the school year 2015-2016.

Throughout these years, the course provided an opportunity to conduct several studies on the teaching of algorithms and programming in secondary education. The research focuses on exploring students' misunderstandings about a) variables, b) loops, c) arrays, d) subprograms and at the same time explores teaching approaches of specific algorithmic issues [11], [6], [12]-[18].

Despite existing research and accumulated experience, algorithmic teaching remains a complex issue. The research highlights issues and difficulties that students have in terms of syntactic, conceptual, and strategic knowledge [19]. According to a recent study, educators' confidence a) in dealing with students' misconceptions in the context of programming and b) on how to teach algorithmic concepts, can affect students' misconceptions [20].

For this purpose, various approaches are used in the teaching of the course. One of these approaches, includes tasks that are given to students and they are asked to describe and explain a code snippet. Students are asked to read a ready-made code and determine what the function of the algorithm. With questions such as: "What is the function of the algorithm?" students have the opportunity to develop algorithmic writing skills and reduce their own mistakes when developing algorithms [21]. Another approach that is widely used both during the teaching and during the national examination of the students, is the virtual execution of the algorithm with an appropriate table where the students record the variables and the values that these variables receive during the execution of the code. This approach gives students the opportunity to study the code, to explain it and to understand better the algorithmic concepts [22]. A third approach is to use half-finished code snippets, where students are asked to fill in the blanks according to the problem. Another approach is

the tasks that is given to students with a ready-made code snippet or a full program and they are asked to find the errors and debug it. Research has shown that debugging is an important practice that contributes in many ways to student knowledge and skills improvement [23]. Utilizing the available programming environments in the computer lab, students have the opportunity to overcome their misconceptions. Using ready-made programming environments, which highlight the elements of a program (such as commands, operators, input, output) in different colors and also allow partial execution of the program, students have the opportunity to experiment, reduce syntax errors, monitor run a program and, ultimately, improve their skills [24].

In this context, the educators' technological pedagogical content knowledge is important. A study by [14], showed that the content knowledge and technology knowledge of computer science educators who teach the course "Informatics" are high (average 4.38/5 and 4.16/5 respectively) and at the same time, it seems that these educators are less confident with their pedagogical content knowledge and their technological content knowledge (average 3.51/5 and 3.68/5 respectively). The results of recent research show that computer science educators often show limited understanding of student misconceptions [25], [19].

Considering all the above, in the next section an attempt will be made to present the research and its results. The results focus on the approaches concerning algorithmic teaching and the computer laboratory use.

IV. The Design and Approach of the Research

This paper presents a part of the research designed to study and record the views, attitudes, and approaches of educators who are teaching the course "Informatics". For this purpose, the positivist paradigm of research was chosen and the conduct of a quantitative research, which attempts to explain how IT educators teach the course "Informatics". A digital questionnaire (as the research tool) was created which given to educators and mainly included closed-ended questions. At the end of each section of questions, participants had the opportunity to write their views on an open-ended question.

The questionnaire aimed to explore several issues related a) the way the course is taught, b) the use of the computer laboratory during the course, c) the assessment techniques used by educators and d) the profile of the participants in the research. The results come from the data analysis of 1127 questionnaires. The research sample was obtained by convenience sampling. Considering that the total number of computer science educators who teach the course is 1437, the participation in relation to the population exceeded 78%. At the same time, the research sample is representative concerning the gender of the population [26].

V. Research Results

In the following sections the demographics data of the sample will presented, followed by descriptive statistics on how the educators that participated in this research teach algorithmic concepts, data structures, as well as modular programming. In addition, some statistically significant differences that occur will be presented. Then, the results of the research related to the use of the computer laboratory during the teaching of the course will be presented.

A. Sample

Female participants make up 35% of the sample, compared to 39% of all female computer science educators who teach the course "Informatics" in public and private schools in Greece [26].

TABLE II. Working Experience in Teaching "Informatics"

Years of experience	Percentage %
1 - 3	7
4 - 6	20
7 - 9	38
More than 9	35

Most educators in the sample (58%) holds a bachelor's degree, 38% have a master's degree and the rest have a PhD (4%). 76% of the participants are under 50 years old. Their working experience in teaching the course "Informatics" is presented in Table II.

B. Programming Language Instead of a Hypothetical Greek Language Specifically for the Course

One of the first questions asked to educators was about the teaching of algorithmic and programming concepts with a programming language, instead of a hypothetical Greek programming language designed specifically for the course. Educators were asked to answer if they disagree or agree with the following statement "I would prefer to teach algorithmic concepts with a known programming language". As shown in Fig. 1, 36.4% stated that they would like to use for the course a known programming language.

There is a statistically significant negative correlation between the experience in teaching the course and the preference for teaching with a known programming language ($X^2 = 11.03$, DF = 3, $p = 0.012$). Educators with more years of working experience prefer to a lesser extent to teach the algorithmic concepts with a known programming language, unlike the rest.

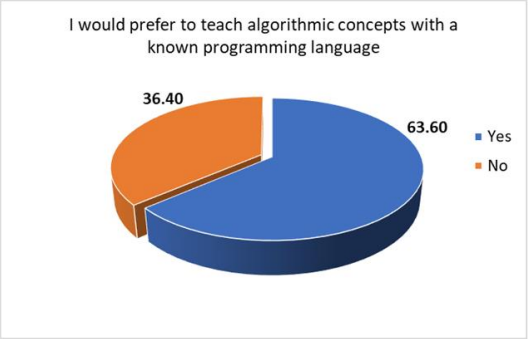

Figure 1. Preferences on using programming language to teach algorithmic concepts

C. Teaching Approaches

As shown in Table I, the first structure that the educator is required to negotiate with his/her students is the sequence structure. Educators were asked to determine whether they agree or disagree with the statement "I start teaching the sequence structure by executing algorithms and I continue by developing algorithms."

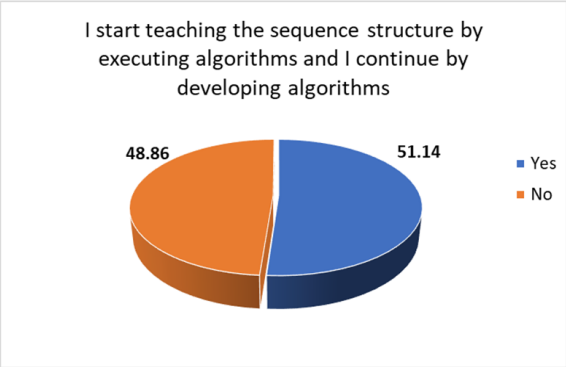

Figure 2. Teaching approaches: Sequence structure

More than half of the educators (51.4%) start teaching the sequence structure by executing an algorithm and then ask their students to work on developing algorithms with sequence structure (see Fig. 2). This approach is quite interesting since the advantages of virtual algorithm execution in terms of understanding algorithmic concepts have already mentioned [22].

In the selection structure the instructions urge the educators to teach the selection commands in the order of if...endif, if...else...endif, if...elseif...else...endif and nested selection commands. Educators were asked to determine the order in which they teach the commands of the selection structure (see Fig. 3).

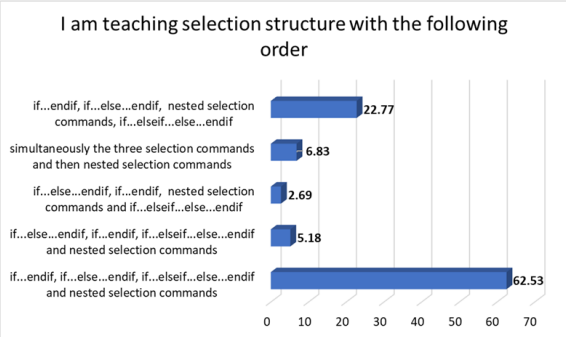

Figure 3. Teaching approaches: Selection structure

A large percentage of educators follow the teaching instructions. In the student book the if...elseif...else...endif command is approached in two ways and more specifically with the if...elseif...else...endif command and the Case command. This approach also explains the large percentage of the first column, since 2 in 10 educators choose to teach the if...elseif...else...endif command as the last command of selection structure.

In the loop structure, the instructions urge educators to approach the three commands in the order: While...do, Repeat...until, For...endfor. Educators were asked to determine the order in which they teach the commands of the loop structure (see Fig. 4). According to the results, only one in two educators follows the teaching instructions. 50% of educators uses a different teaching approach to teach loop commands, with 1 in 4 educators starts teaching with the For…endfor command.

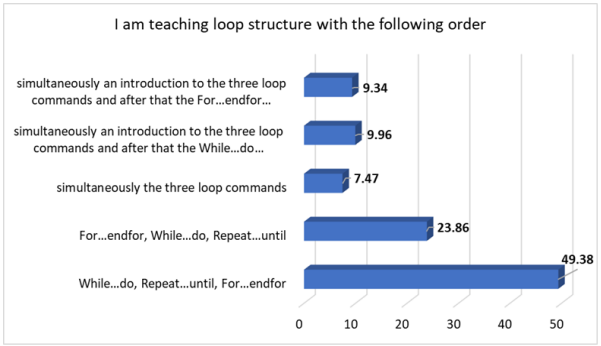

Figure 4. Teaching approaches: Loop structure

In the questions that explored the order in which they teach the selection structure, and loop structure, there is a positive correlation between those who follow the proposed instructions ($X^2 = 45.82$, df = 16, $p < 0.05$). It seems that educators who follow the suggested instructions for selection structure also follow them for the loop structure.

According to the teaching instructions, the arrays are approached in the following order: one-dimensional arrays, two-dimensional arrays and then the operations of searching and sorting. Educators were asked to answer whether they agree or disagree with the statement: "I start teaching one-dimensional arrays, searching and sorting and then two-dimensional arrays and searching and sorting". The vast majority of educators (93%) complete the one-dimensional arrays, searching and sorting operations and continue with two-dimensional arrays (see Fig. 5).

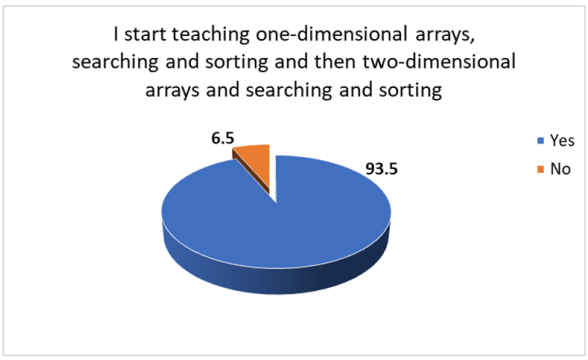

Figure 5. Teaching approaches: Arrays

The modular programming according to the teaching instructions, are approached firstly with a simple problem that is solved using procedures and functions.

The problem is analyzed in subproblems, then the necessary procedures and functions are developed and then the main program that calls the subprograms is

presented. After that, a virtual execution of the program takes place in order to complete the negotiation of the issue. Educators were asked to answer if they agree or disagree with the statement "In modular programming, I first teach program execution with subprograms and then I develop programs with subprograms". Two out of three educators use a teaching approach that starts with virtual execution of a program with subprograms and then they proceed to the development of programs with subprograms (see Fig. 6).

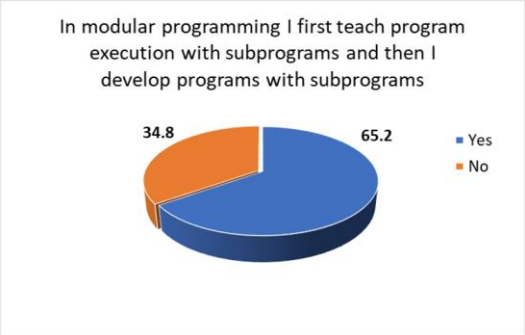

Figure 6. Teaching approaches: Modular programming

In the modular programming, the approach suggested in the student book includes: a complete example of a program, followed by functions and procedures, and the teaching is completed with an example of how the parameters pass and return from / to the program to / from the procedure. Thus, educators were asked to choose the order in which they teach subprograms.

Fig. 7 highlights the variety of teaching approaches followed by educators. The percentages are shared and show the relative autonomy of the educators when choosing the teaching approach for modular programming. One explanation for the small adoption of the instructions is that this chapter was added to the curriculum in the school year 2002-2003, that is three years after the start of the course. It seems, then, that the educators had established their beliefs on how to teach the lesson and therefore chose with relative autonomy the way they would teach this new chapter.

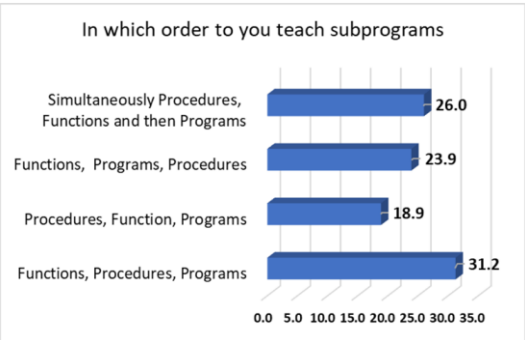

Figure 7. Teaching approaches: Subprograms presentation

D. Computer Lab and Classroom Use

The frequency of use of computer laboratory and the way it is utilized, were further explored. Regarding the utilization of the computer laboratory, it emerged that 57.4% of the educators use it during the teaching of the course (see Fig. 8).

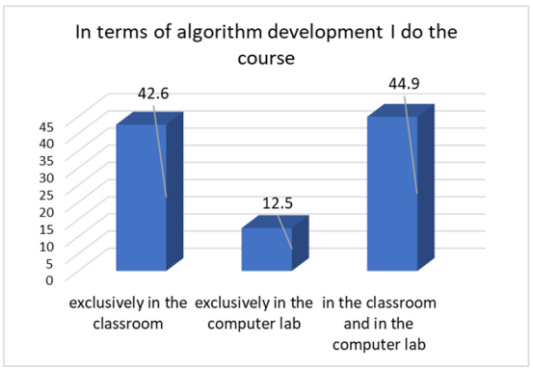

Figure 8. Computer lab and classroom use

There is a statistically significant difference between males and females educators concerning computer laboratory use. Males appear to choose to a greater extent the use of computer laboratory than females, who largely choose exclusively the classroom ($X^2 = 7.28$, DF = 2, p = 0.26).

Although it is suggested that the course be taught in the computer lab, 4 out of 10 educators do not use it during the school year. It is worth mentioning that 39% of these educators do not use a computer even in the classroom as a medium of teaching (see Fig. 9). Therefore, 15% of the educators who participated in this research teach programming exclusively on the whiteboard, without giving students the opportunity to work with a computer and take advantage of the possibilities of the medium.

Of the educators who do not use the computer laboratory at all, 76.8% accept that the use of technological tools has pedagogical value and 70.1% accept the view that educational software can enrich the teaching of the course. At the same time, 62.2% of the educators who do not use the computer laboratory at all, claim that they have educational material for teaching at the computer laboratory, but 93.3% state that they do not use it because the available teaching hours are not enough to add laboratory hours to the course.

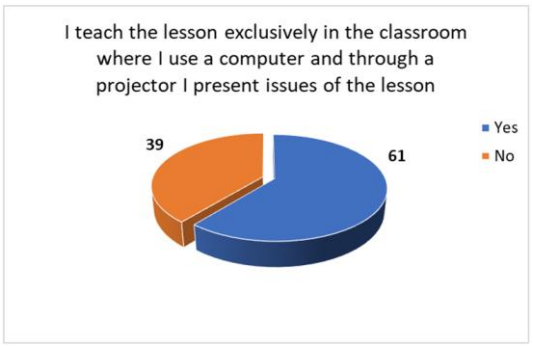

Figure 9. Using whiteboard with projector

At the same time, 33.5% of educators adopt the statement that the use of the computer laboratory has no

practical value, since the examination is done on paper. Classroom management issues and teaching strategies required in the computer laboratory do not seem to be an obstacle in order to use it. Only 3 out of 10 educators say that they prefer the classroom because they can better manage students than the computer lab. In addition, 20% claim that teaching the course in the classroom while developing algorithms, give them more teaching strategies than the computer lab.

It is also interesting to note that although teachers believe that their students have the ability to work on computers, as only 1 in 10 teachers say that students do not know how to use computers well, a percentage of 24.4% say that they can not to do the lesson in the computer lab, as the situation in it is not satisfactory for conducting a laboratory lesson or there are not enough computers for all students (40.2%).

Many of the problems are adequately summed up in a colleague's comment: “The main reason I do the lesson in the classroom and not in the lab is the lack of time. In a teaching period in the computer laboratory, we can solve 1 exercise whereas in the classroom usually 2 to 3. Also, usually the number of children in a classroom is 22 and the laboratories have 12 computers, so it is very difficult for everyone to work alone... ".

The above statement "... In a teaching period in the computer laboratory we can solve 1 exercise whereas in the classroom usually 2 to 3 ...", together with the classroom management issues and teaching strategies highlight issues related to the possible training needs of the educators in this field and in particular with their training needs in technological and pedagogical content knowledge [14].

On the other hand, there is a percentage of educators (12.5%) who use the laboratory exclusively for teaching the course. Of these educators, 79.2% say that the main reason for using the laboratory is the appropriate educational software that contributes to learning. In addition, 6 out of 10 educators point out that in the laboratory they have tools (projector, computer, and appropriate presentation software) that facilitate teaching. In this context, it is interesting that 62.5% of the educators who use exclusively the computer laboratory, claim that "they save time by teaching in the laboratory".

TABLE III. DISTRIBUTION OF TEACHING TIME BETWEEN CLASSROOM AND COMPUTER LABORATORY

Concept	Classroom	Same time	Computer Lab
Sequence structure	74.6 %	16.8 %	8.6 %
Selection structure	71.1 %	20.8 %	8.1 %
Loop structure	64.2 %	26.6 %	9.2 %
Arrays	68.2 %	20.8 %	11.0 %
Modular programming	62.4 %	19.1 %	18.5 %

The third group of educators uses the laboratory and the classroom together (44.9%). It is interesting how they divide the time between the laboratory and the classroom. Table III shows the distribution of time according to educators' answers. As shown in Table III, in all algorithmic concepts (sequence, selection, loops, arrays and modular programming) the use of the classroom is much greater than the use of the computer lab.

The utilization of both the classroom and the computer lab, show that this group of educators has a high pedagogical and technological content knowledge. More specifically, 92.5% of the educators who follow the mixed model believe that there are sections of the course that are suitable to be taught in the computer laboratory and other sections where they have to negotiate in the classroom. In contrast to educators who use the computer laboratory exclusively, educators who use both the classroom and computer laboratory do not consider that they save time in the computer laboratory in the same degree (48.6% compared to 62.5% of the colleagues in the exclusive use of the computer laboratory).

Summarizing the above data, it seems that during the mixed use of classroom and computer laboratory, educators balance, on the one hand, the need to prepare students for the national exams and, on the other hand, the need to deepen students in algorithmic and computational thinking. using appropriate educational software.

VI. DISCUSSION

The teaching of algorithms and programming in secondary education that includes national exams is an interesting feature of modern curricula. It is important, that many countries and many educational systems around the world, try to integrate relevant courses in their curricula which are aimed at developing algorithmic thinking, but also preparing future citizens who can deal professionally with computer science and programming. This effort will have a significant impact on future societies.

This research highlights that educators follow a variety of approaches to teach programming concepts. On the one hand, they prepare students for exams at the national level, which is a competitive process, since the grade that students will receive determines the entrance to university departments. On the other hand, they attempt to build students' algorithmic thinking. It seems that the stronger the algorithmic thinking, the better the performance of the students. However, there are cases where teachers prepare students with a behavioral approach, with the sole aim of good performance in exams, although the literature shows that the development of algorithmic thinking requires a constructive teaching approach and therefore active participation. of the students [12].

However, as the research showed, the available time and the material that the teacher needs to cover in each lesson brings significant difficulties in carrying out teaching interventions with constructive practices. For this reason, there is a reduced use of computer laboratories.

Another important finding relates to teachers' perspectives on the use of computer labs. It turns out that out of 42.6% of teachers who do not use the laboratory, a significant percentage consider it a waste of time, as students are not examined nationally in a programming

environment (14.2% of all participants). In addition, 22% of all participants believe that the lesson is delayed when it takes place in the laboratory and therefore fewer activities take place than could be done in the classroom. Contrary to this view, people who use the lab (either exclusively or with mixed approaches) believe that they save time in the lab.

The above finding, in combination with the findings related to the teaching approaches, lead to the investigation of the possible educators' training needs in matters of pedagogical content knowledge, technological content knowledge and technological pedagogical content knowledge. In addition, there is a need for communities of practice and learning for the course, in order to transform the educators' perspective within a framework of collaboration and development. This is also related to the preparation of future educators by university departments. It is an interesting issue to explore how future educators are prepared to teach algorithmic and programming lessons in secondary education.

All the above shows the crucial role of targeted actions that can lead to the transformation of educators' vision and experience, so that the latter can build a teaching framework that will favor the active participation of students.

VII. Conclusion

In recent years, there is an intense discussion about changes in the curricula, in order to emerge material with specifications that will enhance algorithmic and computational thinking. Computer science educators, who are called to teach algorithmic and programming courses in secondary education, have taken on an important role that may benefit society in the coming decades. The development of algorithmic thinking, computational thinking and programming knowledge may equip students with necessary skills for the future. Therefore, educators with many years of service, need to enhance their pedagogical content knowledge (PCK) and their technological content knowledge (TCK).

Moreover, educators who have informal "apprenticeship" in the subject as they have attended this course as students in secondary education need a) to be involved in communities of practice concerning the way of teaching algorithms and b) to be trained on issues of pedagogical and technological content knowledge in higher education.

In this way they will be able to establish views, perceptions, and attitudes about the way in which they will transform their own knowledge, so that they can facilitate the learning of their students. In this way, they will prepare the next generation, so that to be literate in programming and be able to read, use and explain code.

Conflict of Interest

The author declares no conflict of interest.

Author Contributions

SD conducted the research, analyzed the data, wrote the paper; ...; all authors had approved the final version.

References

[1] F. B. Flórez, R. Casallas, M. Hernández, A. Reyes, S. Restrepo, and G. Danies, "Changing a generation's way of thinking: Teaching computational thinking through programming," *Review of Educational Research*, vol. 87, no. 4, pp. 834-860, 2017.

[2] A. Luxton-Reilly, I. Albluwi, B. A. Becker, *et al*., "Introductory programming: a systematic literature review," in *Proc. 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, 2018, pp. 55-106.

[3] R. P. Medeiros, G. L. Ramalho, and T. P. Falcão, "A systematic literature review on teaching and learning introductory programming in higher education," *IEEE Transactions on Education*, vol. 62, no. 2, pp. 77-90, 2018.

[4] N. S. Papaspyrou and S. Zachos, "Teaching programming through problem solving: The role of the programming language," in *Proc. Federated Conference on Computer Science and Information Systems*, 2013, pp. 1545-1548.

[5] A. Vakali, E. Giannopoulos, N. Ioannidis, C. Koilias, K. Malamas, Y. Manolopoulos, and P. Politis, *Applications Development in a Programming Environment*, 1999. (in Greek).

[6] S. Siozou, N. Tselios, and V. Komis, "Effect of algorithms' multiple representations in the context of programming education," *Interactive Technology and Smart Education*, 2008.

[7] N. Avouris, M. Margaritis, and V. Komis, "Modelling interaction during small-group synchronous problem-solving activities: The Synergo approach," in *Proc. ITS 2004 Workshop on Designing Computational Models of Collaborative Learning Interaction*, 2004, pp. 13-18.

[8] E. Koffmann and T. Brinda, "Teaching programming and problem solving: The current state," *Informatics Curricula and Teaching Methods*, Springer, Boston, MA., pp. 125-130, 2003.

[9] V. Dagdilelis, G. Evangelidis, M. Satratzemi, V. Efopoulos, and C. Zagouras, "DELYS: A novel microworld-based educational software for teaching computer science subjects," *Computers & Education*, vol. 40 no 4, pp. 307-325, 2003.

[10] M. Giannakos, P. Hubwieser, and N. Chrisochoides "How students estimate the effects of ICT and programming courses," in *Proc. 44th ACM Technical Symposium on Computer Science Education*, 2013, pp. 717-722.

[11] V. Dagdilelis, M. Satratzemi, and G. Evangelidis, "Introducing secondary education students to algorithms and programming," *Education and Information Technologies*, vol. 9, no 2, pp. 159-173, 2004.

[12] M. Kordaki, M. Miatidis, and G. Kapsampelis, "A computer environment for beginners' learning of sorting algorithms: Design and pilot evaluation," *Computers & Education*, vol. 51, no 2, pp. 708-723, 2008.

[13] S. Doukakis, C. Koilias, N. Adamopoulos, and P. Giannopoulou, "Computer science teachers' in-service training needs and their technological pedagogical content knowledge," in *World Summit on Knowledge Society*, Springer, Berlin, Heidelberg, 2011, pp. 311-316.

[14] S. Doukakis, A. Psaltidou, A. Stavraki, N. Adamopoulos, P. Tsiotakis, and S. Stergou "Measuring the technological pedagogical content knowledge (TPACK) of in-service teachers of computer science who teach algorithms and programming in upper secondary education," in *Proc. Readings in Technology and Education: Proceedings of ICICTE*, 2010, pp. 442-452.

[15] E. Vrachnos and A. Jimoyiannis, "Design and evaluation of a web-based dynamic algorithm visualization environment for novices," *Procedia Computer Science*, vol. 27, pp. 229-239, 2014.

[16] S. Psycharis and M. Kallia, "The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving," *Instructional Science*, vol. 45, no. 5, pp. 583-602, 2017.

[17] E. Seralidou and C. Douligeris, "Learning programming by creating games through the use of structured activities in secondary education in Greece," *Education and Information Technologies*, pp. 1-40, 2020.
[18] P. Koutsakas, C. Karagiannidis, P. Politis, and I. Karasavvidis, "A computer programming hybrid MOOC for Greek secondary education," *Smart Learning Environments*, vol. 7, no. 1, pp. 1-22, 2020.
[19] Y. Qian and J. Lehman, "Students' misconceptions and other difficulties in introductory programming: A literature review," *ACM Transactions on Computing Education (TOCE)*, vol. 18, no. 1, pp. 1-24, 2017.
[20] Y. Qian, S. Hambrusch, A. Yadav, S. Gretter, and Y. Li, "Teachers' Perceptions of Student Misconceptions in Introductory Programming," *Journal of Educational Computing Research*, vol. 58, no. 2, pp. 364-397, 2020.
[21] D. Teague and R. Lister, "Programming: Reading, writing and reversing," in *Proc. Conference on Innovation & Technology in Computer Science Education*, 2014, pp. 285-290.
[22] A. Vihavainen, T. Vikberg, M. Luukkainen, and M. Pärtel, "Scaffolding students' learning using test my code," in *Proc. 18th ACM Conference on Innovation and Technology in Computer Science Education*, 2013, pp. 117-122.
[23] R. McCauley, S. Fitzgerald, G. Lewandowski, L. Murphy, B. Simon, L. Thomas, and C. Zander, "Debugging: A review of the literature from an educational perspective," *Computer Science Education*, vol. 18, no. 2, pp. 67-92, 2008.
[24] C. Kelleher and R. Pausch, "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers," *ACM Computing Surveys (CSUR)*, vol. 37, no 2, pp. 83-137, 2005.
[25] N. C. Brown and A. Altadmri, "Novice Java programming mistakes: Large-scale data vs. educator beliefs," *ACM Transactions on Computing Education (TOCE)*, vol. 17, no. 2, pp. 1-21, 2017.
[26] Hellenic Statistical Authority, "General lyceums, population, units, staff," 2019.

Spyridon Doukakis earned his PhD in Education & Digital Technologies from the University of the Aegean and he completed his postdoctoral research at Ionian University in the field of Neuroeducation. He is a researcher at Ionian University and educator of Mathematics and Computer Science at Pierce-The American College of Greece. He has worked as a Special Counselor for Teacher Training at the Institute of Educational Policy for the Hellenic Ministry of Education. He has co-authored more than 100 papers in international journals and conferences as well as book chapters. He has received a Fulbright scholarship and has been awarded with the Harvard Prize Book Teacher Award