


RESEARCH ARTICLE

WILEY

C Tutor usage in relation to student achievement and progress: A study of introductory programming courses in Portugal and Serbia

Luís Alves¹ | Dušan Gajić² | Pedro Rangel Henriques³ |
 Vladimir Ivančević²  | Vladimir Ivković² | Maksim Lalić² | Ivan Luković² |
 Maria João Varanda Pereira⁴ | Srđan Popov² | Paula Correia Tavares⁵

¹Departamento de Informática e Comunicações, Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Bragança, Bragança, Portugal

²Department of Computing and Control Engineering, Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia

³Departamento de Informática, Escola de Engenharia, Universidade do Minho, Braga, Portugal

⁴Centro de Investigação em Digitalização e Robótica Inteligente (CeDRI), Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Bragança, Bragança, Portugal

⁵Departamento de Engenharia Informática, Instituto Superior de Engenharia do Porto, Instituto Politécnico do Porto, Porto, Portugal

Correspondence

Vladimir Ivančević, Department of Computing and Control Engineering, Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad 21000, Serbia.

Email: dragoman@uns.ac.rs

Funding information

Foundation for Science and Technology, Portugal; Ministry of Education, Science, and Technological Development, Serbia, Grant/Award Numbers: 451-03-1924/2016-09/53, III-44010, 451-03-68/2020-14/200156

Abstract

Previous research studies on introductory programming courses in engineering education in Portugal and Serbia have indicated that although high motivation and high expectations seem to be reported by students, many students may fail the course. This prompted a further inquiry into student attitudes, behavior, and achievement, and it also led to the introduction of C Tutor, a widely known program visualization tool, into courses in both countries. As a result, in the present study, self-reported student achievement (grades), self-reported student progress (knowledge improvement and confidence), and self-reported usage and helpfulness of C Tutor were investigated. Anonymous data about students and their experience in the course, which also included the usage of C Tutor, were collected in a survey in Portugal and Serbia. Quantitative methods, including descriptive statistics, clustering, statistical testing of independence, and partial correlation analysis, were applied in analyses of survey data. The distribution of grades differed between the two countries, but overall attitudes were similar. Various uncovered patterns involving student attitudes and usage of C Tutor may serve as a starting point for new research studies.

KEYWORDS

C Tutor, introductory programming, program visualization, student confidence, student knowledge

1 | INTRODUCTION

Learning programming may be a challenging task, even for students at the university level. Within the scope of a bilateral research project between Portugal and Serbia, it

was observed that pass rates were rather low in some introductory programming courses [5] and that enrolled students generally reported low entrance knowledge [6]. These negative trends might have far-reaching implications, as many struggling students may drop out at the

beginning of their studies, or even if they manage to complete introductory courses, the level of attained knowledge may be barely sufficient for advanced courses. Such negative outcomes could be disquieting for programming teachers, especially as students in the two countries may start their programming courses with high expectations and strong motivation [6]. For these reasons, the research within the bilateral project focused on deeper understanding of students and exploration of teaching and learning methods in introductory programming.

The initial positive attitude of students towards their studies may not be surprising. In a survey involving first-year engineering students from three higher education institutions in Europe, students generally expressed their confidence in the successful completion of the first year of the studies, which may be in disagreement with the actual drop out rates at the same institutions [15]. In a study focusing on introductory computer science (CS), at the end of a course, students commonly reported that they were confident in their ability to write a program, but an additional inquiry into retention rates uncovered lower retention among students without previous programming experience [55]. Such diverse findings may point to a complex set of associations between student confidence, entrance knowledge, knowledge improvement during a course, and actual academic achievement.

The observed problems in introductory programming courses may also be partially attributed to the peculiarities of the domain, as novice programmers generally encounter various difficulties while learning programming [1]. The development of computer-assisted instruction has resulted in a wide set of resources being available to both students and teachers, who are sometimes unaware of many freely available software tools that may support learning and teaching. The usefulness of various software tools in programming instruction has been evaluated in numerous studies, often with positive results [48]. Despite not being always successful in practice, programming assistance tools may be promising instruments when devising educational intervention [26], in particular, tools for visualization in programming [40].

As a result, within the bilateral project, a web-based software tool for program visualization (PV), named C Tutor [19], was introduced into introductory programming courses in Portugal and Serbia. C Tutor is a service for visualizing execution of C programs, and it is a part of the Python Tutor web environment, which was created in 2010 with support for Python only and has had at least 3.5 million users from more than 180 countries [17–19]. This tool was chosen because it offers solid code visualization capabilities for the C language, which is the language used in the courses considered in the project, and it is both mature and freely available online.

In the present study, student attitudes, behavior, and achievement were investigated in the context of introductory programming courses in Portugal and Serbia. The following questions were examined:

- What are the overall levels of student achievement?
- How much do students believe that their programming-related knowledge has improved?
- How confident do students feel in performing programming-related activities?
- In which setting do students use C Tutor and how helpful do they consider it to be when learning programming?

The investigation of these questions was conducted primarily using quantitative methods on data that were self-reported by students in a survey. The collected survey data were also explored in search of any major patterns that may be related to one or more of the posed questions, especially to those regarding C Tutor.

2 | RELATED WORK

The level of confidence that students possess might affect their learning and progress. The notion of self-efficacy, which was developed by Bandura [7], may be especially influential in this regard. A positive correlation between self-efficacy and performance in a single course was detected in a study involving adult learners [14]. In another correlation analysis, a solid positive relationship was observed between the self-efficacy of university students at three different levels and their academic performance, as represented by grade point average (GPA) [46]. A similar, but weaker, relationship regarding self-efficacy and GPA was also reported for undergraduate students from various departments [28]. In a regression analysis of data about engineering students, GPA was positively related to self-efficacy [22].

However, issues concerning student achievement in programming may be related to the nature of programming. Fundamental programming concepts could be too abstract and hard to grasp for first-year students in introductory CS courses [31,36]. For instance, in a study involving teachers with experience in programming teaching, the concepts of recursion, pointers, loops, and functions were considered the most problematic [16]. Moreover, in the same study, teachers also pointed out the significance of “making programming visual” for students [16]. On the basis of the assumption that some students might benefit from visual representations of concepts while building their mental models [35], plenty of tools for algorithm visualization (AV) and PV have emerged [33,45].

AV tools are primarily focused on illustrating abstract concepts of the algorithm itself while leaving out technical details of a particular implementation [20]. There are several resources offering animated visualization for various well-known algorithms [9,33,45,53,54]. Also, plenty of AV tools are designed for some specific subtypes of algorithms and concepts in CS theory. There are tools covering specific topics in the area of data structures [10,11], branch and bound strategies [56], regex and automata [4], sorting [29], and expressions [27].

PV tools are more focused on illustrating runtime execution of some actual software implementation [8]. They tend to provide the user with an insight into the execution flow and the content of memory handled by a program during its execution [8]. Some PV tools support only one programming language, such as Jeliot3 and jGrasp for Java [13,37]. Others, like ViLLE [42] and LIVE [10], support multiple programming languages. Python Tutor [17] initially supported only Python but was later extended with similar services for other common languages: C Tutor, C++ Tutor, Java Tutor, JavaScript Tutor, Ruby Tutor, and TypeScript Tutor [19].

Some AV and PV tools are designed as web platforms [20,33,42,45], for which users only need a browser and active Internet connection, whereas others are standalone solutions, which may be used offline when installed on the user's machine [11,13,37]. There are programming languages, such as JAWA and Alice2 [12,41], that are developed just to allow the user to quickly create animations of runtime executions.

In many high schools and universities, teachers are introducing these kinds of tools into their teaching practice, hoping to improve students' learning [25,34,43]. There are numerous such tools, and the choice of whether visualization tools should be introduced into teaching practice, and, if yes, which tools should be included, depends on the context. Researchers suggest that lecturers should keep track of the effects of these tools on the learning process and try to adapt and improve the usage of these tools in their own learning environments [45,52].

In a study comparing PV tools, Python Tutor was preferred by participants when visualizing problems related to parameter passing or object programming [3]. However, in a study focused on bug fixing, participants generally preferred TraceDiff over Python Tutor [49]. Omnicode is based on Python Tutor and it expands ideas behind PV and live programming by always displaying all runtime values for each execution step, which may be of help when solving programming problems and explaining programs in Python [24].

Researchers have also worked on PV tools for C, which resulted in systems such as Bradman [47], VINCE

[44], SeeC [21], and INGINious-C-Tutor [38]. PlayVisualizerC, which is a more recent web-based tool for visualizing C programs, may help users to more accurately and more promptly answer some questions about program execution, as opposed to SeeC or when not using any visualization tool at all [23].

An investigation of introductory programming courses in Portugal and Serbia revealed that certain courses had low pass rates [5]. A subsequent inquiry into initial student knowledge and attitudes as potential reasons behind low achievement uncovered that, in addition to low entrance knowledge of programming, students reported high motivation and had many expectations regarding their courses [6]. Such circumstances may be conducive to the adoption of a PV tool and the intervention attempted in the present study. Given their motivation and expectations, these students may readily embrace the tool, especially if they perceive that it helps them to adopt basic programming concepts and start programming. Moreover, such development might make students more engaged in the course and positively affect low pass rates.

3 | METHODOLOGY

3.1 | Educational context

The present study considered only introductory programming courses in the context of higher education. In total, two courses from the academic year 2018/2019 were analyzed: a course from a study program organized at a higher education institution in Portugal and a course from a study program organized at a higher education institution in Serbia. Both higher education institutions have a dominant technical orientation, and the study programs to which the considered courses belong are engineering programs with a strong emphasis on computing and informatics. Both courses are among the introductory programming courses in their respective programs and mandatory for all students enrolled.

Within the study program in Portugal, there are two introductory programming courses. At the end of these course units, the student is expected to be able to design solutions and implement C programs that solve small/medium/high complexity problems. For this, the student must apply concepts of imperative programming in the C programming language, coding function-based structured programs to manipulate data structures. The student must also be able to use an integrated development environment including the debugging tool to get successful solutions. The syllabus of these courses contains C

language topics that are commonly taught in introductory programming at the university level. In this respect, it is similar to the syllabus of the course in Serbia and many other courses worldwide.

Within the study program in Serbia, the syllabus of the analyzed course starts with algorithms, their formalization and complexity and algorithmic approach to problem-solving. It then proceeds with the introduction to structured programming using the C programming language. The second part of the course is focused on abstract data types and linear and nonlinear data structures, with their sample implementations in the C language.

3.2 | C Tutor application

Within the analyzed courses in Portugal and Serbia, the C Tutor tool was introduced into teaching activities in 2018/2019, and it has also been used in the subsequent academic year. C Tutor was utilized as an auxiliary tool in the course. While performing computer exercises for the course, all major examples were visualized using C Tutor. During the explanation of basic programming concepts of the C programming language by example, teaching assistants used C Tutor to visualize the following concepts:

- Variables: variable types and values;
- Control flow: programming structures describing selection and iteration;
- Function calls and local variables: tracing parameters and return values of functions;
- Pointers: pointer referencing and dereferencing; and
- Heap: heap content in the context of dynamic memory allocation.

C Tutor was primarily used to facilitate understanding of program execution and basic concepts of computer memory. Students were also encouraged to use the tool while practicing programming individually.

3.3 | Student survey

A student survey was conducted in Portugal and Serbia during the academic year 2018/2019. Collected survey data included self-reported data about overall course achievement, improvement in programming-related knowledge, confidence in performing programming-related activities, usage of C Tutor, and self-reported helpfulness of C Tutor when learning programming.

In each country, anonymous questionnaires in the English language were administered during the summer

semester and students were expected to provide requested information about their participation in a programming course conducted in the previous semester. There were 80 respondents from Portugal and 30 respondents from Serbia.

The questionnaire items that are related to the present study are presented in Table 1. Response sets of some items differed between the two countries. The response set of item P2, which refers to student grade, had to be adjusted, as different grading systems and exam policies were used in Portugal and Serbia.

The survey was conducted as a part of a series of surveys on introductory programming across various courses in Portugal and Serbia. The scope of the survey was considerably broader than the scope of the present study. For this reason, the present study did not investigate all of the factors and courses featured in the survey. Factors that are not directly related to student participation and experience within the considered programming course were excluded from the analysis. As the present study explored patterns related to the usage of C Tutor, courses with a relatively low number of students familiar with C Tutor were also excluded from the investigation.

3.4 | Data preparation

For each item listed in Table 1, responses were recoded into a matching set of numeric values. Recoding of a response into a numeric value was performed by preserving only the initial numeric part of the response text. For analytical purposes, additional variables describing individual respondents were derived on the basis of the recoded item responses. These variables include *Practice*, *Grade*, *KnowledgeImprovement*, *Confidence*, *TutorUsed*, and *TutorHelpfulness*.

Variable *Practice* denotes self-reported student practice activity regarding programming outside the classroom. It is based on the recoded responses to item P1, which were not subjected to any additional transformations.

Variable *Grade* serves as a uniform description of self-reported student grades across Portugal and Serbia. As Portugal and Serbia use different grading systems in education, this variable was introduced to facilitate a comparison of self-reported student achievement between the two countries. Variable *Grade* was derived from responses to item P2, as presented in Table 2. It features Levels 1–6, where Level 1 denotes the absence of grade, Level 2 denotes the lowest passing grade level, and Levels 3–6 denote additional passing grade levels in the ascending order.

TABLE 1 Key items from the survey questionnaire used in Portugal (PT) and Serbia (RS)

Items related to programming course and introductory programming	
Code	Item with responses
P1	Did you practice programming for the course outside the classroom? 1. No 2. Yes, but only before assessments 3. Yes, occasionally, in spare time 4. Yes, on a regular basis
P2 ^{PT}	Have you passed the course and, if yes, what is your grade? 1. No, I need to repeat the course in the next academic year 2. No, but I have the chance to pass in a special examination period 3. Yes, my grade is in range [10 ; 11] 4. Yes, my grade is in range [12 ; 13] 5. Yes, my grade is in range [14 ; 15] 6. Yes, my grade is in range [16 ; 17] 7. Yes, my grade is in range [18 ; 20]
P2 ^{RS}	Have you passed the course and, if yes, what is your grade? 1. No, I need to repeat the course in the next academic year 2. No, I need to take the exam in order to pass the course 3. No, but I have enough points to get a passing grade 4. Yes, my grade is 6 5. Yes, my grade is 7 6. Yes, my grade is 8 7. Yes, my grade is 9 8. Yes, my grade is 10
P3	I have improved my knowledge in algorithms. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
P4	I have improved my knowledge in programming languages. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
P5	I have improved my knowledge in software development. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
P6	I am now feeling confident in understanding functions and code snippets. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree

TABLE 1 (Continued)

Items related to programming course and introductory programming	
Code	Item with responses
P7	I am now feeling confident in understanding whole programs. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
P8	I am now feeling confident in writing code by myself. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
P9	I am now feeling confident in optimizing my code. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
Items related to C Tutor usage	
Code	Item with responses
T1	I used this tool during the introductory programming course. 1. No 2. Yes, but only inside the classroom 3. Yes, but only outside the classroom 4. Yes, both inside and outside the classroom
T2	This tool helped me understand data structures. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
T3	This tool helped me understand control flow. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
T4	This tool helped me identify and fix errors in code. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree
T5	This tool helped me in my “trial & error” method when developing solutions. 1. Strongly disagree 2. Disagree 3. Neutral 4. Agree 5. Strongly agree

(Continues)

TABLE 2 Derivation of variable *Grade* from responses to item P2

Level of variable <i>Grade</i>	Matching P2 responses in Portugal	Matching P2 responses in Serbia
1	1. No, I need to repeat the course in the next academic year 2. No, but I have the chance to pass in a special examination period	1. No, I need to repeat the course in the next academic year 2. No, I need to take the exam in order to pass the course 3. No, but I have enough points to get a passing grade
2	3. Yes, my grade is in range [10 ; 11]	4. Yes, my grade is 6
3	4. Yes, my grade is in range [12 ; 13]	5. Yes, my grade is 7
4	5. Yes, my grade is in range [14 ; 15]	6. Yes, my grade is 8
5	6. Yes, my grade is in range [16 ; 17]	7. Yes, my grade is 9
6	7. Yes, my grade is in range [18 ; 20]	8. Yes, my grade is 10

Variable *KnowledgeImprovement* describes self-reported improvement in student knowledge of algorithms, programming languages, and software development. Its individual values were derived by averaging individual responses to items P3–P5.

Variable *Confidence* describes self-reported student confidence when performing certain programming-related activities. Its individual values were derived by averaging individual responses to items P6–P9.

Variable *TutorUsed* explains whether a student used C Tutor, as reported in the survey. It was based on responses to item T1. It features value *No*, which corresponds to the first T1 response (response 1), and value *Yes*, which corresponds to the remaining T1 responses (responses 2, 3, and 4).

Variable *TutorHelpfulness* describes how helpful C Tutor was to a student while learning programming, as reported in the survey. It denotes self-reported helpfulness of the tool as support in concept understanding, error resolution, or solution development. The individual variable values were derived by averaging individual responses to items T2–T5.

3.5 | Data analysis

The inspection of self-reported student grades was performed separately for students from Portugal and Serbia. The empirical cumulative distribution function was determined for variable *Grade*. Grade class was modeled as variable *GradeClass* with values low (*Class 1*) and high (*Class 2*), based on the value of variable *Grade*. Grade levels that were characteristic of the low-achieving half of students in a course were labeled *Class 1*, whereas *Class 2* was the label for the remaining grade levels, that is, the grade levels of the high-achieving half. The grade level threshold between *Class 1* and *Class 2* was selected in a manner that favors split of responses into two groups of equal or nearly equal size.

The mean values of the derived variables, *KnowledgeImprovement*, *Confidence*, and *TutorHelpfulness*, were visualized separately for Portugal and Serbia. Variables *KnowledgeImprovement* and *Confidence* were also used as a basis for identifying student clusters in Portugal and Serbia. K-means was used as the clustering algorithm, whereas the number of clusters was set to two. The distribution of different grade classes between the clusters was presented visually, with added jitter to reduce overlap between symbols in plots.

Patterns involving variables *GradeClass* and *TutorUsed* were examined using contingency tables for Portugal and Serbia. The association between the two variables was evaluated using two-tailed Fisher's exact test. Moreover, indicators of overall student performance within the course in Portugal were calculated using anonymous aggregated data and then compared between the year when C Tutor was not used and the following year in which C Tutor got introduced into the course. Patterns involving variable pairs that are composed of variable *TutorHelpfulness* and one of the variables *Grade*, *Confidence*, and *KnowledgeImprovement* were examined using partial correlation. For each considered variable pair, the partial correlation was calculated by applying Spearman's method while using variable *Practice* as a control variable. This analysis was performed separately for responses from Portugal and Serbia, and it considered only responses for which the value of variable *TutorUsed* was *Yes*.

The *p* value level .05 was set as a threshold for determining statistical significance. Preparation and analysis of data were conducted using the R environment for statistical computing [51] with its system libraries and the ppcor and psych libraries, which are available in the CRAN (Comprehensive R Archive Network) repository [50]. RStudio [39] was used as a software environment for the creation and execution of R program scripts.

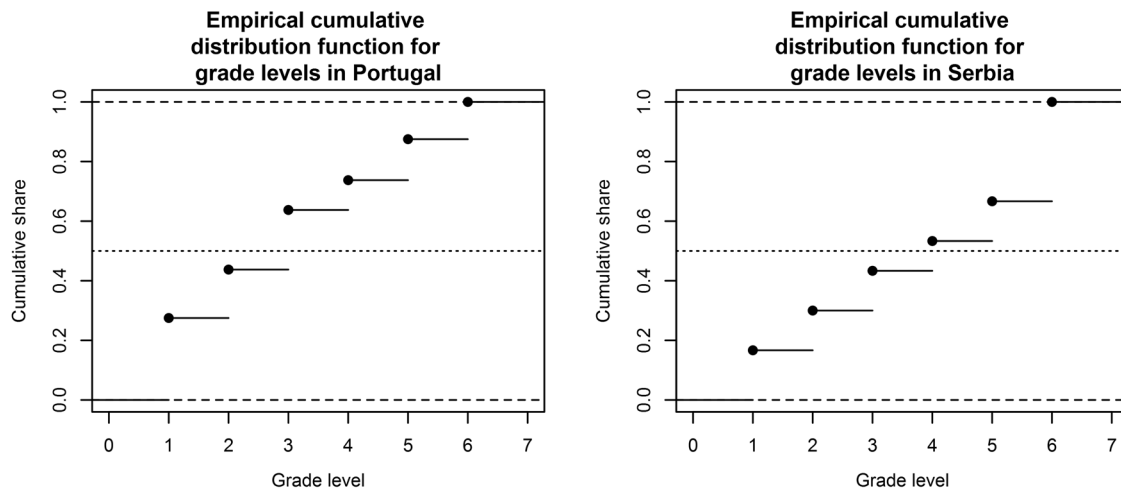


FIGURE 1 The distribution of variable *Grade* within the course in Portugal (left) and within the course in Serbia (right)

4 | RESULTS AND DISCUSSION

4.1 | Achievement

In Figure 1, the distribution of values of variable *Grade* is presented separately for Portugal ($n = 80$) and Serbia ($n = 30$). Student pass rates somewhat differ between the courses: 72.5% in Portugal versus 83.3% in Serbia. This finding generally agrees with the previously observed difference in passed/assessed ratios in introductory programming between the two countries [5].

A prominent difference between the two distributions of self-reported grade levels is the share of extreme grade levels. Within the course in Portugal, the lowest grade level (Level 1), which generally denotes the lack of a grade, is the most common. On the contrary, within the course in Serbia, the highest grade level (Level 6) is the most common. In general, student achievement appears to be higher in Serbia, but, due to different contexts of the two courses, any conclusions regarding observed differences would be highly speculative without further investigation.

As grade distribution differed between the courses, grade threshold, which divides grade levels into *Class 1* and *Class 2* of variable *GradeClass*, was set to Level 2 for the course in Portugal and Level 4 for the course in Serbia. These grade levels were selected because their cumulative shares were closest to 0.5.

4.2 | Attitudes and C Tutor

Self-reported responses about knowledge improvement, confidence, and helpfulness of C Tutor were analyzed only for respondents who reported using C Tutor in Portugal ($n = 62$) and Serbia ($n = 22$). The reliability of

the questionnaire scales was evaluated with respect to the calculated values of Cronbach's α , which are given in Table 3. In all instances, Cronbach's α is above .70, which is generally considered acceptable.

Mean values of *KnowledgeImprovement*, *Confidence*, and *TutorHelpfulness* are presented in Figure 2. These values do not considerably differ between the two courses. They are always between 3 (response *Neutral*) and 4 (response *Agree*), but generally closer to the latter (except for *TutorHelpfulness* within the course in Serbia). This finding indicates that there is a positive trend in self-reported knowledge improvement and self-reported confidence, and, to some extent, in self-reported tutor helpfulness as well.

The share of C Tutor users is comparable between the analyzed courses: 77.5% in Portugal versus 73.3% in Serbia. In both countries, the most frequent mode of usage is using the tool only inside the classroom: 41.3% in Portugal versus 56.7% in Serbia. The use of the tool outside the classroom, whether as the sole mode of usage or combined with usage inside the classroom, is more common in Portugal (36.3%) than in Serbia (16.7%).

TABLE 3 Cronbach's α values for variables *KnowledgeImprovement*, *Confidence*, and *TutorHelpfulness* within the course in Portugal (C Tutor users only) and within the course in Serbia (C Tutor users only)

Variable	Cronbach's α	
	Course in Portugal (C Tutor users only)	Course in Serbia (C Tutor users only)
<i>KnowledgeImprovement</i>	.75	.78
<i>Confidence</i>	.86	.84
<i>TutorHelpfulness</i>	.84	.90

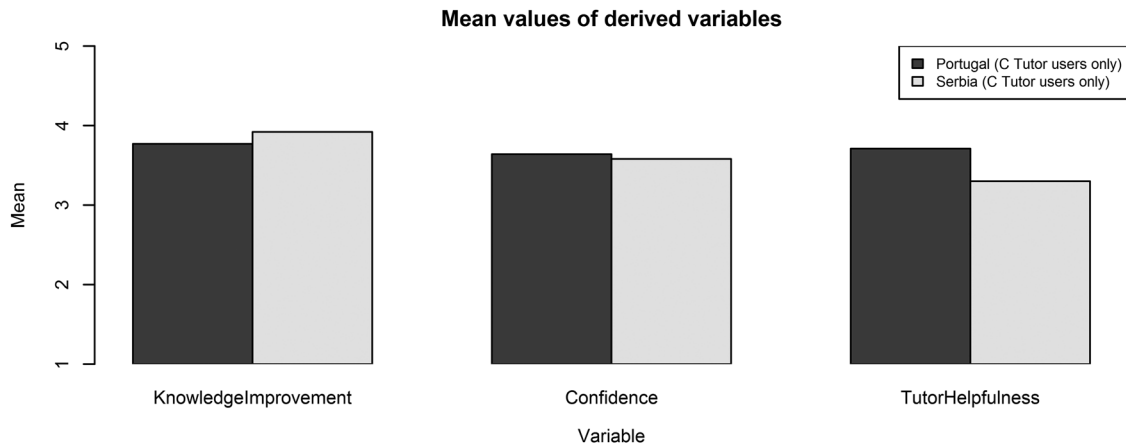


FIGURE 2 Mean values of variables *KnowledgeImprovement*, *Confidence*, and *TutorHelpfulness* within the course in Portugal (C Tutor users only) and within the course in Serbia (C Tutor users only)

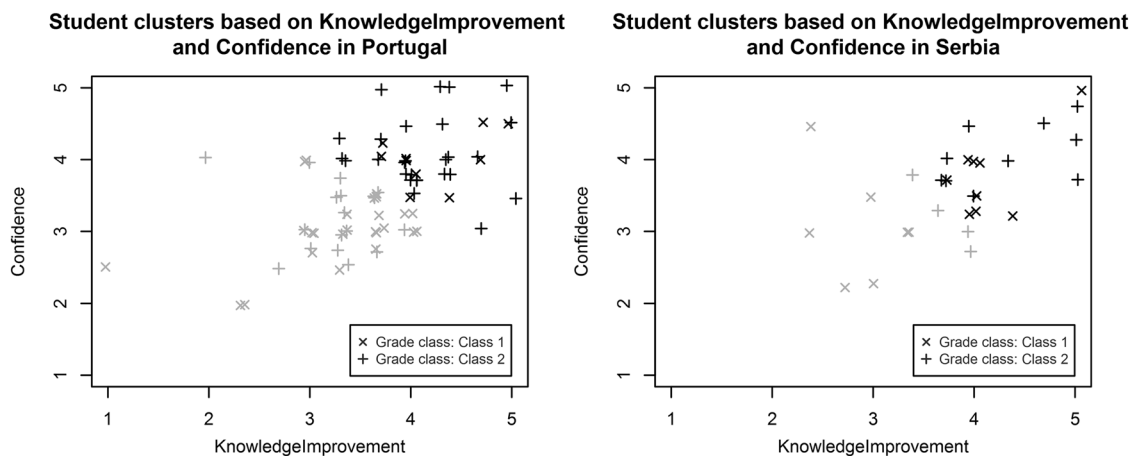


FIGURE 3 Student clusters based on variables *KnowledgeImprovement* and *Confidence* within the course in Portugal (left) and within the course in Serbia (right). Each symbol corresponds to a student, whereas symbol shape denotes self-reported student achievement transformed into grade class ('x'Class 1 and '+'Class 2). For each course, there are two clusters discernible by symbol color

4.3 | Major patterns in data

In Figure 3, student clusters are presented separately for Portugal and Serbia. Members of different clusters may be discerned by symbol color, whereas the grade class of each member may be identified by symbol shape. There appears to be a common trend as variables *KnowledgeImprovement* and *Confidence* tend to increase jointly for both countries. Moreover, the higher grade class is more common in the cluster that corresponds to higher values of variables *KnowledgeImprovement* and *Confidence*.

Tables 4 and 5 are contingency tables for the variable pair *GradeClass-TutorUsed* within the course in Portugal and within the course in Serbia, respectively. The application of two-tailed Fisher's exact test to each contingency table yielded statistically significant results: $p = .033$ for data from Portugal and $p = .039$ for data from Serbia. In both analyzed courses, the share of C Tutor users is higher in the group of high-achieving students,

TABLE 4 The distribution of students within the course in Portugal with respect to variables *GradeClass* and *TutorUsed*

		<i>TutorUsed</i>	
		No	Yes
<i>GradeClass</i>	Class 1	12	23
	Class 2	6	39

TABLE 5 The distribution of students within the course in Serbia with respect to variables *GradeClass* and *TutorUsed*

		<i>TutorUsed</i>	
		No	Yes
<i>GradeClass</i>	Class 1	7	9
	Class 2	1	13

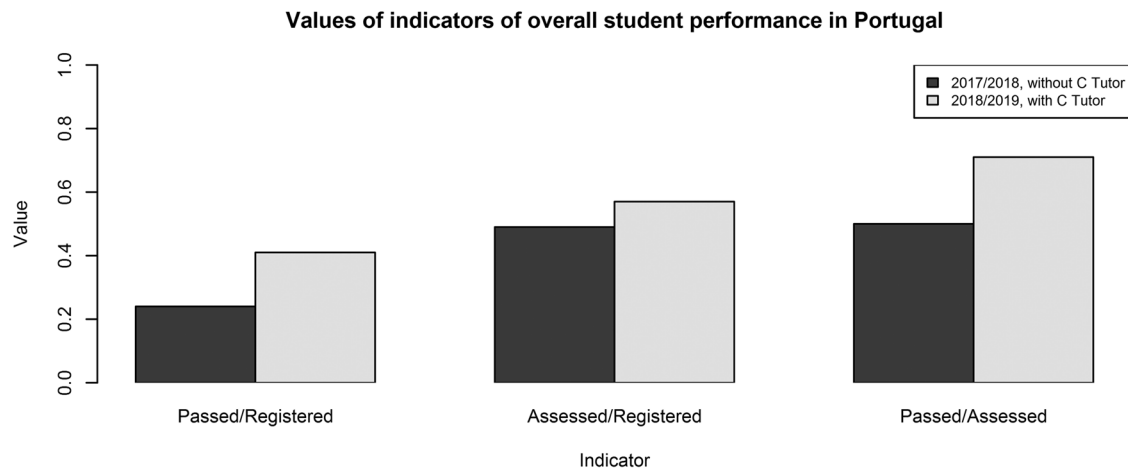


FIGURE 4 Values of indicators of overall student performance within the course in Portugal in 2017/2018, when C Tutor was not used, and in 2018/2019, when C Tutor was used

that is, students whose grades are labeled as *Class 2*. Additional research would be needed to examine how tutor usage might be related to achievement and whether it could have a positive influence.

In Figure 4, there is a comparison of overall student performance within the course in Portugal between two successive years. Anonymous aggregated data were used to calculate the values of three performance indicators for both years. The indicators were defined as ratios involving the numbers of registered, assessed, or passed students. C Tutor was not used in the first year, but it was introduced into the course in the second year, for which a considerable increase in student performance across all three indicators may be observed. Although the observed trend is positive, further investigation would be required

to more carefully evaluate if C Tutor has an impact on overall student performance.

Patterns between self-reported helpfulness of C Tutor and different indicators of self-reported progress and self-reported achievement in introductory programming were evaluated using Spearman's partial correlation with self-reported practice behavior acting as a control variable. Tables 6 and 7 present results of partial correlation analysis concerning data from Portugal and Serbia, respectively. A similar trend regarding correlation coefficient values may be observed in both courses. For each course, there is a positive correlation between *TutorHelpfulness* and *KnowledgeImprovement*, which is the strongest one. This is followed by a weaker positive correlation between *TutorHelpfulness* and *Confidence*, whereas the weakest

TABLE 6 A pairwise partial correlation for responses from Portugal (C Tutor users only) involving variables *KnowledgeImprovement*, *Confidence*, and *Grade* as the first variable and *TutorHelpfulness* as the second variable, with variable *Practice* as a control variable

Variable 1	Variable 2	Control variable	Partial correlation	p Value
<i>KnowledgeImprovement</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	.34	.007
<i>Confidence</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	.17	.197
<i>Grade</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	.08	.558

TABLE 7 A pairwise partial correlation for responses from Serbia (C Tutor users only) involving variables *KnowledgeImprovement*, *Confidence*, and *Grade* as the first variable and *TutorHelpfulness* as the second variable, with variable *Practice* as a control variable

Variable 1	Variable 2	Control variable	Partial correlation	p Value
<i>KnowledgeImprovement</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	.44	.048
<i>Confidence</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	.35	.123
<i>Grade</i>	<i>TutorHelpfulness</i>	<i>Practice</i>	-.15	.507

correlation is the one between *TutorHelpfulness* and *Grade*. The partial correlation is statistically significant only for the variable pair *TutorHelpfulness–Knowledge-Improvement*. Visualization tools could be considered a subtype of simulation tools for which, in general, there is evidence that they may positively affect learners' knowledge and confidence [2]. As a result, a potentially beneficial impact of C Tutor usage on improvement in knowledge might not be surprising. However, confidence improvement in programming-related activities could be affected not only by the extent of knowledge of fundamental concepts in CS but also by other factors, such as usage of development tools, a clear understanding of language syntax [32], and practical experience [30]. Any confirmation of the influence of C Tutor usage on any complex interaction between improvement in knowledge, confidence, and grade would demand further research involving more respondents, more variables, and more advanced analysis and research methods.

5 | CONCLUSION

The present study was conducted primarily using self-reported student data with the goal of providing some insight into student attitudes, behavior, and achievement in selected introductory programming courses in Portugal and Serbia. It was discovered that derived pass rates matched some of the rates observed in previous research. Distribution of grades reported by students differed between the two countries. In Serbia, the highest grade level was the most common among the six derived grade levels, which was in contrast with the distribution of grades in Portugal. The levels of self-reported improvement in programming knowledge and self-reported confidence regarding programming were quite similar between the two countries and generally in the positive zone. The initial response and preliminary results regarding the introduction of the C Tutor tool into programming courses may seem promising. According to the data, in both countries, C Tutor was used more by high-achieving students and the dominant mode of usage was inside the classroom only. Various other patterns involving self-reported usage or self-reported helpfulness of the tool were found, such as a positive correlation between self-reported knowledge improvement and self-reported helpfulness of C Tutor with self-reported practice as a control variable. Nonetheless, a systematic evaluation of those patterns would demand a new study following a different design.

C Tutor appears to be an effective tool that may be used in a straightforward manner, even by beginners in programming. Teachers may relatively easily incorporate the tool in the modern computer classroom. Program examples may be prepared before class and demonstrated in class or

shared with students, who can experiment with the tool. A greater reliance on tools such as C Tutor may help teachers to reduce the need for the manual drawing of program execution diagrams.

A potential risk concerning the impact of C Tutor may be excessive reliance of students on the tool in their programming activities. In the present study, according to the collected data, some students did not use the tool, and, among those who did, a large number of students restricted the use to the classroom only. These findings could indicate that the overreliance on C Tutor was probably not very common among the students in the analysed courses. However, in different environments, students may behave differently. The risk of overreliance on C Tutor or similar tools could probably be limited by adequate assessment policies. Assessments in introductory programming courses are often organized in a controlled environment, which excludes the usage of external resources. This could prevent students from accessing non-essential programming tools and also motivate them to develop better programming skills. Moreover, many visualizations of program execution have an educational purpose and serve to illustrate concepts from introductory programming. Students tend to master introductory concepts and may also start using debuggers, so the need for a visualization tool might diminish and even completely vanish.

The study has certain limitations. Its primarily descriptive and correlational nature may lessen confidence in provided interpretations of the results. The used survey data were actually self-reported by the participants, which might raise some concerns regarding objectivity and data veracity. Moreover, the size of the collected data set may be another restrictive factor. In a search for global trends in data, smaller sample sizes may be problematic when utilizing more demanding analysis methods. However, the size of the data sample is practically limited by the size of the considered course and the share of students willing to use a specific software tool in their programming education.

As the tool offers visualization of various data structures, execution of instructions one by one, and collaborative programming, it could be an asset to students and teachers alike. Its visual character may support students to persist in their journey throughout introductory programming with greater enthusiasm and dedication. The reported findings of relationships between C Tutor usage and various aspects of achievement and progress in introductory programming education may be regarded as positive in some respects, especially given the international scope of the study. A stronger and wider effect of introducing a new tool into teaching may have been expected. However, mixed results regarding the use of similar tools have been observed by some researchers [26,48]. In general, the manner in which a tool is used, as well as the extent of usage, could be important

and should probably be considered when planning proper integration of the tool into teaching. An experiment would be needed to more closely investigate whether, and potentially how much, C Tutor may help in improving concrete individual skills that are commonly taught in introductory programming.

It may be also argued that other factors may contribute much more to student improvement and especially to academic achievement. At present, the obtained findings may serve as a starting point for further inquiries into the benefits of specialized educational technologies for programming education and their concrete impact on knowledge attainment and confidence. Another track of inquiry may be directed at student motivation, especially at understanding which teaching techniques and technology resources could help teachers to raise and preserve student interest in programming.

ACKNOWLEDGMENTS

The research was conducted within the project “Data mining based evaluation of IT teaching practice in Portugal and Serbia” in the scope of the Program of Scientific and Technological Cooperation between the Government of the Portuguese Republic and the Government of the Republic of Serbia. It was supported by the coordinator of the Program in the Portuguese Republic, the Ministry of Education and Science, as represented by the Foundation for Science and Technology (Portugal–Serbia Cooperation Programme), and by the coordinator of the Program in the Republic of Serbia, the Ministry of Education, Science, and Technological Development (Serbia–Portugal Cooperation Programme, Grant 451-03-1924/2016-09/53). The research was also supported by the Ministry of Education, Science, and Technological Development of the Republic of Serbia within the projects “Intelligent systems for software product development and business support based on models” (Grant III-44010) and “Innovative scientific and artistic research from the FTS (activity) domain” (Grant 451-03-68/2020-14/200156). The authors are thankful to all the respondents in the survey and all the persons who assisted in the administration of the survey.

ORCID

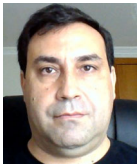
Vladimir Ivančević  <https://orcid.org/0000-0002-1106-7265>

REFERENCES

1. K. Ala-Mutka, *Problems in learning and teaching programming—A literature study for developing visualizations in the Codewitz-Minerva Project*, Codewitz Needs Analysis, 2004, available at https://www.cs.tut.fi/~edge/literature_study.pdf
2. A. A. Alanazi, N. Nicholson and S. Thomas, *The Use of simulation training to improve knowledge, skills, and confidence among healthcare students: A systematic review*, Internet J. Allied Health Sci. Pract. **15** (2017), no. 3, 2.
3. S. Alhammad, S. Atkinson and L. Stuart, *The role of visualization in the study of computer programming*, Proceedings of the 27th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2016), Cambridge, UK, 2016, pp. 5–16.
4. M. A. M. Al-Nakhal and S. S. Abu Naser, *Adaptive intelligent tutoring system for learning computer theory*, Eur. Acad. Res. **4** (2017), no. 10, 8770–8782.
5. L. Alves, et al., *A comparison of introductory programming courses between Portugal and Serbia*, Proceedings of the 8th International Conference on Applied Internet and Information Technologies (ICAIIIT 2018), Bitola, Republic of Macedonia, 2019, pp. 88–93.
6. L. Alves, et al., *Student entrance knowledge, expectations, and motivation within introductory programming courses in Portugal and Serbia*, Proceedings of the 47th SEFI Annual Conference 2019, Budapest, Hungary, 2019, pp. 1354–1363.
7. A. Bandura, *Self-efficacy: Toward a unifying theory of behavioral change*, Psychol. Rev. **84** (1977), no. 2, 191–215.
8. S. Bentrat and D. Meslati, *Visual programming and program visualization—Toward an ideal visual software engineering system*, ACEEE Int. J. Inform. Technol. **1** (2011), no. 3, 43–49.
9. *Build your knowledge of data structures through visualizations and practice, OpenDSA*, available at <https://opensa-server.cs.vt.edu/>
10. A. E. R. Campbell, G. L. Catto and E. E. Hansen, *Language-independent interactive data visualization*, ACM SIGCSE Bull. **31** (2003), no. 1, 215–219. <https://doi.org/10.1145/792548.611972>
11. L. Christiawan and O. Karnalim, *AP-ASD1: An Indonesian desktop-based educational tool for basic data structure course*, Jurnal Teknik Informatika dan Sistem Informasi **2** (2016), no. 1, 21–30. <https://doi.org/10.28932/jutisi.v2i1.422>
12. S. Cooper, W. Dann and R. Pausch, *ALICE: A 3-D tool for introductory programming*, J. Comput. Sci. Coll. **15** (2000), no. 5, 107–116.
13. J. H. Cross, D. Hendrix and D. A. Umphress, *JGRASP: An integrated development environment with visualizations for teaching java in CS1, CS2, and beyond*, Proceedings of 34th Annual Frontiers in Education (FIE 2004), Savannah, GA, 2004, pp. 1466–1467.
14. M. De Fátima Goulão, *The relationship between self-efficacy and academic achievement in adults' learners*, Athens J. Educ. **1** (2014), no. 3, 237–246.
15. T. De Laet et al., *Confidence in and beliefs about first-year engineering student success: Case study from KU Leuven, TU Delft, and TU Graz*, Proceedings of the 45th SEFI Annual Conference 2017, Azores, Portugal, 2017, pp. 894–902.
16. A. Gomes et al., *A teacher's view about introductory programming teaching and learning—Portuguese and Macanese perspectives*, Proceedings of 2017 IEEE Frontiers in Education Conference (FIE 2017), Indianapolis, IN, 2017, pp. 1–8.
17. P. J. Guo, *Online python tutor: Embeddable web-based program visualization for CS education*, Proceedings of the 44th ACM Technical Symposium on Computer Science Education, Denver, CO, 2013, pp. 579–584.
18. P. Guo, *Building tools to help students learn to program*, Commun. ACM **60** (2017), no. 12, 8–9.

19. P. Guo, *Python Tutor*, available at <http://pythontutor.com/>
20. S. Halim et al., *Learning algorithms with unified and interactive web-based visualization*, *Olymp. Inform.* **6** (2012), 53–68.
21. M. Heinsen Egan and C. McDonald, *Program visualization and explanation for novice C programmers*, Proceedings of the 16th Australasian Computing Education Conference (ACE 2014), Auckland, New Zealand, 2014, pp. 51–57.
22. C.-L. Huang, *Self-efficacy in the prediction of academic performance of engineering students*, Proceedings of the 2003 ASEE Gulf-Southwest Annual Conference, Arlington, TX, 2003, pp. 1–9.
23. R. Ishizue et al., *PVC: Visualizing C programs on web browsers for novices*, Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE 2018), Baltimore, MD, 2018, pp. 245–250.
24. H. Kang and P. J. Guo, *Omnicode: A novice-oriented live programming environment with always-on run-time value visualizations*, Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST 2017), Quebec City, QC, Canada, 2017, pp. 737–745.
25. O. Karnalim and M. Ayub, *The use of python tutor on programming laboratory session: Student perspectives*, *Kinetik: Game Technol. Inf. Syst. Comput. Netw. Comput. Electron. Control* **2** (2017), no. 4, 327–336. <https://doi.org/10.22219/kinetik.v2i4.442>
26. M. Koorsse, C. Cilliers, and A. Calitz, *Programming assistance tools to support the learning of IT programming in South African secondary schools*, *Comput. Educ.* **82** (2015), 162–178. <https://doi.org/10.1016/j.compedu.2014.11.020>
27. A. Kumar, *Results from the evaluation of the effectiveness of an online tutor on expression evaluation*, Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, New York, NY, 2005, pp. 216–220.
28. Y. Köseoğlu, *Self-efficacy and academic achievement—A case from Turkey*, *J. Educ. Pract.* **6** (2015), no. 29, 131–141.
29. V. Ladislav and S. Veronika, *Algorithm animations for teaching and learning the main ideas of basic sortings*, *Inform. Educ. Vilnius Univ. Inst. Math. Inform.* **16** (2017), no. 1, 121–140. <https://doi.org/10.15388/infedu.2017.07>
30. T. Levine and S. Donitsa-Schmidt, *Computer use, confidence, attitudes, and knowledge: A causal analysis*, *Comput. Hum. Behav.* **14** (1998), no. 1, 125–146.
31. R. Lister et al., *A multi-national study of reading and tracing skills in novice programmers*, *ACM SIGCSE Bull.* **36** (2004), no. 4, 119–150. <https://doi.org/10.1145/1044550.1041673>
32. A. K. Lui et al., *Saving weak programming students: Applying constructivism in a first programming course*, *ACM SIGCSE Bull.* **36** (2003), no. 2, 72–76. <https://doi.org/10.1145/1024338.1024376>
33. A. Majumder, *A taxonomy for animation-aided visualization tools*, Master's thesis, Iowa State University, 2019.
34. S. Maravić Čisar et al., *Effectiveness of program visualization in learning Java: A case study with Jeliot 3*, *Int. J. Comput. Commun. Control* **6** (2011), no. 4, 668–680. <https://doi.org/10.15837/ijccc.2011.4.2094>
35. R. E. Mayer, *Systematic thinking fostered by illustration in scientific text*, *J. Educ. Psychol.* **81** (1989), no. 2, 240–246. <https://doi.org/10.1037/0022-0663.81.2.240>
36. M. McCracken et al., *A multi-national, multi-institutional study of assessment of programming skills of first-year CS students*, *ACM SIGCSE Bull.* **33** (2001), no. 4, 125–140. <https://doi.org/10.1145/572139.572181>
37. A. Moreno et al., *Visualizing programs with Jeliot 3*, Proceedings of the Working Conference on Advanced Visual Interfaces (AVI 2004), Galipoli, Italy, 2004, pp. 373–376.
38. N. Ooghe, *An online C programming tutor*, Master's thesis, Université catholique de Louvain, 2016.
39. *Open source and professional software for data science teams, RStudio*, available at <https://rstudio.com/>
40. A. Pears et al., *A survey of literature on the teaching of introductory programming*, *ACM SIGCSE Bull.* **39** (2007), no. 4, 204–223. <https://doi.org/10.1145/1345375.1345441>
41. W. C. Pierson and S. H. Rodger, *Web-based animation of data structures using JAWAA*, Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education, Atlanta, GA, 1998, pp. 267–271.
42. T. Rajala et al., *VILLE—Multilanguage tool for teaching novice programming*, technical report 827, Turku Centre for Computer Science, Turku, Finland, 2007.
43. T. Rajala et al., *Effectiveness of program visualization: A case study with the ViLE tool*, *J. Inform. Technol. Educ.* **7** (2008), 15–32. <https://doi.org/10.28945/195>
44. G. Rowe and G. Thorburn, *VINCE—An on-line tutorial tool for teaching introductory programming*, *Br. J. Educ. Technol.* **31** (2000), no. 4, 359–369. <https://doi.org/10.1111/1467-8535.00168>
45. C. A. Shaffer et al., *Algorithm visualization: The state of the field*, *ACM Trans. Comput. Educ.* **10** (2010), no. 3, 1–22. <https://doi.org/10.1145/1821996.1821997>
46. R. Shkullaku, *The relationship between self-efficacy and academic performance in the context of gender among Albanian students*, *Eur. Acad. Res.* **1** (2013), no. 4, 467–478.
47. P. A. Smith and G. I. Webb, *The efficacy of a low-level program visualization tool for teaching programming concepts to novice C programmers*, *J. Educ. Comput. Res.* **22** (2000), no. 2, 187–215. <https://doi.org/10.2190/N0VV-0P48-XJ9G-F8WV>
48. J. Sorva, V. Karavirta, and L. Malmi, *A review of generic program visualization systems for introductory programming education*, *ACM Trans. Comput. Educ.* **13** (2013), no. 4, 1–64. <https://doi.org/10.1145/2490822>
49. R. Suzuki et al., *TraceDiff: Debugging unexpected code behavior using trace divergences*, Proceedings of the 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2017), Raleigh, NC, 2017, pp. 107–115.
50. *The comprehensive R archive network*, CRAN, available at <https://cran.r-project.org/>
51. *The R project for statistical computing*, R, available at <https://www.r-project.org/>
52. J. Urquiza-Fuentes and J. Á. Velázquez-Iturbide, *A survey of successful evaluations of program visualization and algorithm animation systems*, *ACM Transactions on Computing Education—Special Issue on the 5th Program Visualization Workshop (PVW'08)*, 2009, pp. 1–21.
53. *Visualising data structures and algorithms through animation, VisuAlgo*, available at <https://visualgo.net/en/>
54. *Visualize your algorithms*, Avy, available at <http://www.algoviz.net/>
55. C. Wilcox and A. Lionelle, *Quantifying the benefits of prior programming experience in an introductory computer science course*, Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE 2018), Baltimore, MD, 2018, pp. 80–85.
56. S. Zumaytis, and O. Karnalim, *Introducing an educational tool for learning branch & bound strategy*, *J. Inform. Syst. En. Bus. Intell.* **3** (2017), no. 1, 8. <https://doi.org/10.20473/jisebi.3.1.8-15>

AUTHOR BIOGRAPHIES



Luís Alves has an MSc in Computer Science from the University of Porto in 2001. He is currently finishing his PhD thesis in the Information Systems and Technologies Doctorate Program at the School of Engineering of the University of Minho. He is a research collaborator member at the ALGORITMI Research Center in the Software-based Information Systems Engineering and Management Group (SEMAG) integrated in the Information Systems and Technologies (IST) line. He is currently an assistant lecturer at the Technology and Management School of the Polytechnic Institute of Bragança at the Informatics and Communications Department, where he teaches different courses in the broader area of programming. His scientific interests and publications concern empirical software engineering, software metrics, software process improvement, project management, and computer-assisted education.



Dušan Gajić is an assistant professor at the Department of Computing and Control Engineering of the University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia. He received his MSc and PhD degrees in Electrical Engineering and Computing from the University of Niš, Faculty of Electronic Engineering, Serbia, in 2009 and 2014, respectively. His research interests include parallel and distributed computing, blockchain, GPGPU, algorithm design, and computer vision. He has authored or co-authored five chapters in international scientific monographs, more than 60 papers published in peer-reviewed journals or presented at scientific conferences, and three technical solutions. He received best paper awards at ETRAN 2013 and DIPP 2015 conferences.



Pedro Rangel Henriques got a degree in Electronics Engineering at FEUP (Porto University), and finished a PhD thesis in Formal Languages and Attribute Grammars at the Minho University, where he also got his Habilitation in 2012. In 1981 he joined the Computer Science Department of the University of Minho, where he is a teacher/researcher. Since 1995 he has been the coordinator of the Language Processing group at the ALGORITMI Research Center. From 1981 until now, he has taught different courses in the broader area of programming: Programming Languages and Paradigms; Compilers; Language Engineering; Grammar

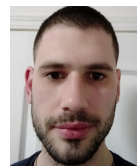
Engineering and Software Analysis and Transformation; Introduction to ICT; etc. He has supervised 17 PhD theses (13 finished), more than 45 MSc theses, and many (>100) graduating trainings/projects, in the areas of: language processing (textual and visual) and structured document processing; code analysis, program visualization/animation, and program comprehension; and knowledge discovery from databases, data-mining, and data-cleaning. He is the co-author of the “XML & XSL: da teoria a prática” book, published by FCA in 2002, has published >15 chapters in books, >35 journal papers, and >100 conference papers, and has been enrolled in 28 R&D projects.



Vladimir Ivančević is working as an assistant professor at the Faculty of Technical Sciences at the University of Novi Sad, Serbia. His research interests are related to Data Science, Artificial Intelligence, Software Engineering, and Databases. He has been involved in projects focusing mainly on application of computer science and informatics across various domains.



Vladimir Ivković is a PhD student and a teaching assistant at the Faculty of Technical Sciences, University of Novi Sad, involved in several undergraduate courses in the field of computer science. He received his BSc and MSc degrees in Computing and Control Engineering from the University of Novi Sad. His research interests include distributed ledger technology, data engineering, and process mining.



Maksim Lalić was born in December 1994 in Novi Sad, Serbia. He received his bachelor's and master's degrees in 2017 and 2018 at the Faculty of Technical Sciences at the University of Novi Sad. He is currently a PhD student at the Faculty of Technical Sciences where he also works as a teaching assistant. He teaches subjects from the area of database systems, data science, and artificial intelligence, which are areas of his interest and further research. He is the co-author of four papers.



Ivan Luković received his diploma degree (five years) in Informatics from the Faculty of Military and Technical Sciences in Zagreb in 1990. He completed his MSc (former Mr, two year) degree at the University of Belgrade, Faculty of

Electrical Engineering in 1993, and his PhD at the University of Novi Sad, Faculty of Technical Sciences in 1996. Currently, he works as a full professor at the Faculty of Technical Sciences of the University of Novi Sad, where he lectures in several Computer Science and Informatics courses. His research interests are related to database systems, business intelligence systems, and software engineering. He is the author or co-author of about 200 papers, four books, and 30 industry projects and software solutions in the area. He created a new set of BSc and MSc study programs in Information Engineering, that is, Data Science, at the Faculty of Technical Sciences. The programs were accredited in 2015.



Maria João Varanda Pereira was born in November 1971 in Braga and received the MSc and PhD degrees in Computer Science from the University of Minho in 1996 and 2003, respectively. She is an integrated member of the Research Centre in Digitalization and Intelligent Robotics (CeDRI) at the Polytechnic Institute of Bragança and a collaborator member of the Language Processing group in the ALGORITMI Research Center at the University of Minho. She is currently a coordinator professor at the Technology and Management School of the Polytechnic Institute of Bragança, and she is a vice president of the same school. As a computer science researcher with 25 years of experience, she is interested and usually supervises Master and PhD students in the following areas: domain specific software development, visualization tools, human-computer interaction, QA systems, data science, machine learning, learning analytics, generation of virtual learning spaces, and computer-assisted education. She is the co-author of 20 articles in journals and 77 articles at international conferences, more than half of which are indexed. She is responsible for or participates in several research projects with international partners from countries such as Poland, Italy, Spain, Romania, Serbia, Slovenia, Argentina, and Brazil.



Srđan Popov was born in August 1969 in Zrenjanin. He completed his basic studies at the Faculty of Technical Sciences in Novi Sad in 1999 with a thesis on “Visualization of XML Documents”. He completed his Master's degree in 2007 with a Master's thesis “Web Services for the Visual Interpretation of Geospatial Data”. He received his

PhD in 2011 on “Application of Service Model for Integrating Heterogeneous Data in the Analysis of Risk of Events with Catastrophic Consequences”. Since 2000 he has been working at the Faculty of Technical Sciences, University of Novi Sad, where he has participated in the realization of several projects. He was a doctoral student at the United Nations University in Bonn, Germany. As a guest lecturer, he has participated in the realization of several UNITED NATIONS UNIVERSITY (UNU-EHS) graduate programs. From June 1999 to the present he has been engaged in projects and teaching at the Faculty of Technical Sciences. He has published more than 100 scientific papers and participated in several projects, 30 of which are of particular interest. He has taught numerous courses: Programming Languages and Data Structures; Object-Oriented Programming; Applied Programming; Computer Networks; etc. He teaches Operating Systems and Computer Networks to talented students in special computer classes at the “Jovan Jovanović Zmaj” High School in Novi Sad.



Paula Correia Tavares has been a professor of the Department of Computer Engineering of the School of Engineering (ISEP) of the Polytechnic Institute of Porto (IPP) since 2007. She has a BSc (five years degree) in Electrical Engineering – Power Systems, ISEP, IPP. She finished a PhD in Informatics (2018) at the School of Engineering of the University of Minho with the title “O Impacto da Animação e da Avaliação Automática na Motivação para o Ensino da Programação”. Nowadays she is working in Learning to Code and her main interests are innovative educational systems and solutions, centered in the teaching of computer science. She has authored or co-authored several conference papers. She is a member of the ALGORITMI Research Center, a research unit of the School of Engineering of the University of Minho.

How to cite this article: Alves L, Gajić D, Rangel Henriques P, et al. C Tutor usage in relation to student achievement and progress: A study of introductory programming courses in Portugal and Serbia. *Comput Appl Eng Educ*. 2020;1–14.
<https://doi.org/10.1002/cae.22278>