# A case study in technology-enhanced learning in an introductory computer programming course

**Jhon J. Ramírez-Echeverry, Felipe Restrepo-Calle & Fabio A. González**

National University of Colombia
Bogotá, Colombia

ABSTRACT: In this article, the authors present a case study of an intervention in an introductory programming course aimed at improving the learning experience of students through the use of three different tools: an interactive document format (Jupyter notebooks), an automatic assessment tool (UNCode), and an education-oriented forum platform (Piazza). The study included 100 students from three different sections of the Introduction to Computer Programming course at an engineering school. The study analysed both the use of the tools and the students' perception of them. Usage was evaluated quantitatively using usage statistics reported by the forum and automatic evaluation tools. Student perception was assessed using a survey with closed-ended and open-ended questions. The results of the study show that all three tools had a positive impact on the students' learning experience. In general, the participants found Jupyter notebooks, UNCode and Piazza forums easy to use, and useful to better understand the programming problems to be solved.

Keywords: Computer programming, technology-enhanced learning, computer science education

## INTRODUCTION

The introductory programming course, usually called CS1, is nowadays part of most engineering and science majors. Programming is rewarded as a very important and useful skill, however, learning it could be a hard challenge for many students. In fact, programming courses are usually regarded as difficult [1][2], according to Rogalski and Samurçay *…Acquiring and developing knowledge about programming is a highly complex process. It involves a variety of cognitive activities, and mental representations related to program design, program understanding, modifying, debugging (and documenting)* [3].

Different strategies have been proposed to mitigate this difficulty, and it has been the focus of a large number of research works during more than half a century. This research has been carried out from different perspectives [4]: teaching and learning strategies, assessment, course design and structure, first languages and paradigms, tools and collaborative approaches, among others. Despite the copious research on the area, programming teaching and learning continues to be a highly researched topic, and the permanent technological development presents new opportunities to enhance and innovate the teaching and learning process.

This article presents a case study of an intervention in an introductory programming course, which aims to improve the learning experience of students by involving three different tools: interactive course materials based on Jupyter notebooks that combine text and programs; an automatic assessment tool, UNCode, for automatic grading of programming assignments; and an education-oriented forum platform, Piazza that allows students to ask and answer questions. The main research question guiding this study is: how students perceive the contribution of these tools to their learning process in the computer programming course? The study involved 100 students from three sections of the Introduction to Computer Programming course in an engineering school.

## COURSE METHODOLOGY

Introduction to Computer Programming is a first-year course taught at the Faculty of Engineering of the National University of Colombia (Universidad Nacional de Colombia) for students of different engineering majors. The objective of this course is to foster the development of algorithmic thinking skills by students, as well as to instruct them in the use of programming languages to solve computational problems in a systematic way. In this course students study the basic

elements of problem solving through the design of algorithms and their implementation in the programming language Python. At the end of the course the students are expected to be able to: recognise problems that can be solved by means of an algorithm; design an algorithmic solution, implement the solution using the Python programming language; test and debug programs written in Python.

The course is organised in 12 thematic units:

1) Problem solving with computers;
2) Variables, data types, expressions, I/O;
3) Conditional control structures;
4) Iterative control structures (while);
5) Iterative control structures (for);
6) Functions;
7) Lists;
8) Arrays and tuples;
9) Sorting, dictionaries and sets;
10) Files and exceptions;
11) Graphics and recursion;
12) Introduction to object-oriented programming.

Each course unit is usually covered in one week.

There are two weekly sessions, each one with a different methodological approach. In the first session, lecture session, the teacher presents the corresponding content using different resources, such as slide presentations, interactive notebooks and live coding; in the second session, practical session, students individually solve different programming exercises with the support of an automatic evaluation tool. The course evaluation is based on coding practical programming assignments.

The course is supported by three main tools:

- Jupyter notebooks are interactive Web documents that combine text content, software code, computational output and multimedia resources [5]. All the contents of the course are written in the form of a textbook composed of different Jupyter notebooks. The students can execute the code examples directly inside each notebook, try different inputs, modify the code and visualise its execution using an interactive visualisation tool (Python Tutor) [6].

- UNCode is a platform for automatic assessment of solutions to programming assignments [7]. It is a Web-based application that allows students to build, test and submit their solutions. UNCode provides a set of tools that support this process including: a code linter, a program-visualisation tool (Python Tutor), the possibility of defining their own test cases (custom input), a feedback system with detailed explanation of errors and a system for tracking student progress [8].

- Piazza is a learning management system that provides forums where students and instructors can ask questions and provide answers [9]. Instructors have the ability to moderate the discussion and endorse the correct answers.

CASE STUDY

The participants in this case study were 100 engineering students in the Introductory Computer Programming course. All students used the Python programming language. These students belonged to three different groups of the course during the first semester of 2020. Therefore, convenience sampling was performed since the study was conducted in the groups in which the students were previously enrolled. In addition, participants signed an informed consent before being included in the study in which they were informed about the objectives of this research.

Quantitative data were collected on students' use of the UNCode and Piazza tools. These data were analysed through descriptive statistics. However, Jupyter notebooks usage could not be tracked as these resources could be accessed directly on Google Colaboratory, which is a cloud-based platform provided by Google.

Among the variables corresponding to the use of UNCode are the following: total submissions, which refers to the total number of submissions made by the students to try to solve the proposed programming problems; accepted, which corresponds to the number of submissions graded with the maximum grade (100%); wrong answer, which refers to submissions with errors in the output of the programs that obtain a partial grade between 0.0% and 99.9%; time limit exceeded, which indicates submissions that exceed the maximum execution time proposed for the task (inefficient solutions); and runtime error, which indicates submissions that generate some type of exception during program execution and interrupt its normal execution.

With respect to Piazza, the variables collected were the following: days on-line, which refers to the number of days that students were logged on to the tool; posts, which corresponds to the number of messages proposed by students; answers,

which indicates the number of answers given by students to a question or discussion; follow-ups, which are follow-up questions to encourage debate and discussion; and replies to follow-ups, which refers to the number of comments received for follow-up questions.

Finally, students were inquired about their opinions regarding the usefulness of each of the tools they used to learn to program, i.e. Jupyter notebooks (Notebooks), UNCode and Piazza. The purpose-designed survey contained three Likert-scale statements and three open-ended questions. For each of the Likert statements, the students chose their level of agreement or disagreement according to the following scale: 1) strongly disagree, 2) disagree, 3) somewhat disagree, 4) somewhat agree, 5) agree, and 6) strongly agree. The open-ended questions asked students why they selected the corresponding option from the Likert scale on each statement. Although all students were asked to complete the survey, their participation was voluntary. Also, data were treated anonymously, and students were previously informed that their responses would not affect their grades. A total of 89 students participated in the survey. The three Likert-scale statements were:

1. Notebooks were useful for learning computer programming.
2. UNCode was useful for learning computer programming.
3. Piazza discussions were useful for learning computer programming.

To analyse the answers to the open-ended questions, a thematic analysis was carried out to identify patterns (themes, categories and indicators) in the entire set of qualitative data by searching for repetitions, similarities and differences, metaphors or analogies, transitions, among other elements from the classification of the units of meaning. The basic unit of meaning defined was each of the answers, complete and uncut, given by the students to each of the questions asked. Analyses were also performed on the basis of word lists (counting), key words in context and co-occurrence of words. The process of thematic analysis began with coding in three stages: open, axial and selective codification. Then, from the coding, categories and themes of a higher degree of abstraction were constructed, which gathered senses not noticed in the first glances at the qualitative data.

## RESULTS

Figure 1 presents the obtained results regarding the students' use of UNCode. These correspond to the results in each of the solution attempts to the programming problems by the 100 students. A total of 19,420 submissions were made, where 26.1% corresponded to accepted submissions, 14.8% were classified as runtime errors, 6.9% were submissions classified as time limit exceeded, and 52.2% were classified as wrong answers. These results include all the problems proposed for each of the groups: 55 in group 1, 55 in group 2 and 56 in group 3.

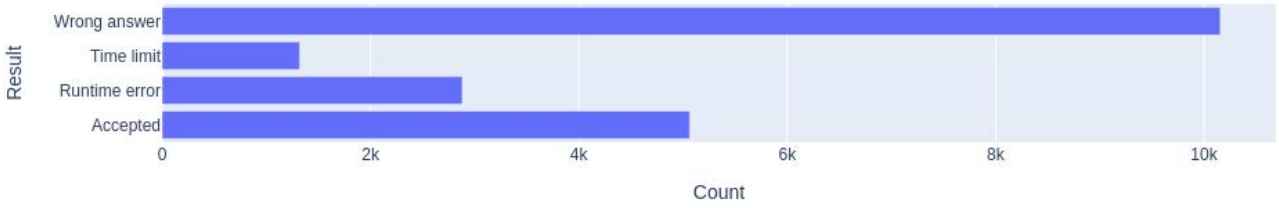


Figure 1: Students' submission results from the solution attempts to the programming problems in UNCode.

Table 1 presents the statistics of student use of Piazza by group. The mean ($\bar{x}$) and standard deviation (SD) of the variables collected are reported: days on-line, posts, answers, follow-ups and replies to follow-ups. The number of students enrolled in each course in Piazza was different from the number of students each group originally had, since the use of this tool was not mandatory. Group 1 included 17 students, group 2 only 12 students, and group 3 included 33 students. In addition, note that in group 3 the use of Piazza was higher because there was a teaching assistant who promoted the participation of students.

Table 1: Students' use of discussion forums in Piazza.

| Group | Days on-line | | Posts | | Answers | | Follow-ups | | Replies to follow-ups | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $\bar{x}$ | SD | $\bar{x}$ | SD | $\bar{x}$ | SD | $\bar{x}$ | SD | $\bar{x}$ | SD |
| 1 | 12.88 | 5.19 | 1.35 | 1.84 | 0.00 | 0.00 | 1.17 | 1.07 | 0.71 | 1.21 |
| 2 | 12.92 | 4.40 | 1.83 | 2.89 | 0.58 | 0.51 | 0.67 | 1.23 | 0.58 | 0.79 |
| 3 | 44.61 | 21.87 | 6.12 | 7.18 | 1.39 | 2.25 | 4.27 | 6.82 | 5.30 | 7.82 |

With respect to the results obtained in the student survey, Figure 2 presents their opinion regarding the usefulness for learning of each of the tools used during the learning process. This figure represents the results obtained in the items of the survey with the Likert scale. Note that the additional option *NA: not applicable* was added to allow students who did not know one of the tools to select this option. The use of Piazza was optional.

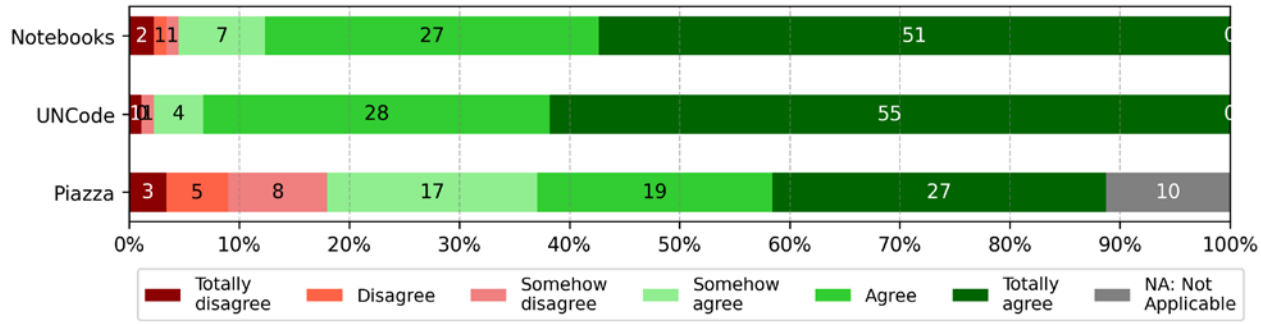


Figure 2: Students' opinions regarding the usefulness for learning of Notebooks, UNCode and Piazza.

As can be seen, results indicate that the majority of students agree to some degree with the usefulness of the three tools for their learning of computer programming: 95.5% in the case of Notebooks, 97.8% with respect to UNCode, and finally, 70.8% concerning Piazza.

The results about the question on the usefulness of the three tools for the learning process, from the analysis of the qualitative data, were as follows:

About Notebooks, three themes were identified, organised in descending order according to the number of references coded in each one: 1) learning tool; 2) teaching tool; and 3) Notebook difficulties.

The first theme includes categories in which the students indicated that Notebooks offered useful learning tools due to their functionalities and the possibilities that favour rigorous conceptual learning, and a thorough understanding of the topics. Among the categories identified were: testing codes in the sense that they allowed students to test codes and their own developments, which led to a better understanding of the functioning of programs, and also the possibility of making corrections and improvements. Likewise, students indicated that the Notebooks became consultation resources in two senses: as a source of information to look for precise conceptual definitions, and as a tool to manage theoretical content that could then be easily consulted, which gives an order to the information that can then be used to solve a problem. Additionally, Notebooks offered a programming environment that facilitated the experience of programming, creating a more orderly workflow for the developments. Notebooks also had the advantage of being available on-line and could be used without the need to run any locally installed program, but only with access to a Web browser. Finally, students noted that Notebooks were easy to use, and that facilitated constant practice of their programming skills.

The second theme, teaching tool, refers to the fact that students perceived that teachers were able to explain programming topics with didactic teaching activities and conceptual definitions illustrated, and well exemplified by the use and integration of Notebooks in the classes. Specifically, two categories were found in the students' opinions: applied examples in the sense that Notebooks allow to see applied examples of codes that are directly related to the class topics, and text cells that are a tool that gives order to the codes written and tested on Notebooks because the inclusion of explanations and titles for the different sections of the commands facilitated the understanding of the particular topics being taught.

The third theme was called Notebook difficulties. This theme groups together difficulties reported by students regarding the use of Notebooks, although there were few opinions referring to these difficulties (nine comments), it is useful to briefly specify the four categories derived: it was noted that it is confusing to use Notebooks, especially in the first contacts, which implies the need to invest significant amounts of time to learn to use them in order to take advantage of their benefits. Similarly, a couple of users reported problems when loading files and folders, i.e. importing items that could then be used directly on Notebooks. The last difficulty reported had to do with a perception of little use in the classes, which prevented students from familiarising themselves with this tool's functions and possibilities.

About UNCode usefulness for learning, from the answers to this question, four main themes were obtained, which were ordered according to the highest number of coded references: 1) general pedagogical achievements; 2) benefits of the platform; 3) UNCode tools; and 4) potential improvements for learning to program.

The first theme was called general pedagogical achievements because it grouped opinions in which students referred to elements of the learning process that benefited from the experience of working with UNCode. Students considered that UNCode allowed them to optimise the evaluation of programming problems because it simplified the presentation of assignments and their assessment. Problem solving was another element mentioned because UNCode allowed them to make different types of controls and checks on the programs they developed, which led them to a better understanding of the code developed, and the possibilities to replicate and apply it in other cases different from those posed in the initial exercises. In other words, UNCode complements and continues in a certain way the instructional process that begins in the synchronous classes, and therefore favours learning and the teaching process itself, since UNCode offers sufficient tools for the refinement and improvement of skills and knowledge.

Another important element was the consideration of the exercises proposed as stimulating exercises because they were sufficiently demanding programming problems to test students' skills and knowledge acquired, which allowed them to complement or continue learning. Additionally, students identified that their programming skills improved thanks to the use of the platform because it favours the acquisition of programming skills that transcend the simple technical use of programming languages. Hence, the instruction received through UNCode favours the development of professional programming skills, such as algorithmic thinking, and the understanding of programming logic as a sequential and systematic process that does not consist of simply replicating commands, but requires a greater degree of planning between a particular objective and the tools available. Finally, another element identified in the students' opinions refers to the fact that UNCode helped them to carry out autonomous learning thanks to the tools it offers, which means that learning continues beyond the shared spaces of the classes, extending to moments in which even the intervention of the teaching staff is minimal or non-existent.

The second theme was called the benefits of the platform and included opinions in which the students explicitly mentioned advantages that they identified that UNCode offered them. The main benefit was that UNCode allowed them to learn about errors, identify and analyse particular errors made when programming. For many students this was the starting point in the process of correcting and improving the programs, and since it is a type of feedback offered by the platform, it then becomes a resource for continued learning. Other benefits were ease of use, formative advice (formative assessment), guidance on the procedure to follow to improve the programs, and the ease of constant practice of their programming skills. Finally, but less frequently, students indicated that UNCode was useful to them thanks to the test cases, on-line availability, and that it became a workspace.

The third theme was called UNCode tools. In this third theme, direct, indirect or generic mentions to the different tools embedded in the platform were grouped: the one with the highest recognition was Python Tutor, which was a highly positive recognition and it was closely linked to the identification and correction of errors, and in general as an added value of whole UNCode that offers very valuable feedback on the programs they developed. Custom input, Linter and unspecified mentions of the tools were less frequent; but which likewise make explicit the tools that students highlighted from UNCode.

Finally, a fourth theme was identified that was called potential improvements for learning to program. Although this theme groups categories with aspects that students considered to be improved in the platform, it grouped a smaller number of opinions, which speaks collaterally of the mostly positive perception of UNCode. Most of these opinions were made in conjunction with the recognition of benefits or achievements that the platform allows. Among the particular categories that refer to aspects identified as problematic, and that deserve to be considered to be improved or corrected in future versions of UNCode are: general failures because the platform was not available on-line due to a temporal server downtime; inflexibility in the validation of the programs because the platform does not accept developments that, although they comply with what was requested in the programming problems, have minor syntax errors that make them appear as incorrect; and insufficient feedback.

About Piazza usefulness for learning, two themes were formed: 1) forum usefulness; and 2) forum difficulties. The reports of forum usefulness duplicated the reports of forum difficulties.

The first theme, forum usefulness, included categories, such as doubt resolution because in the forums the students found answers to doubts that appeared during the resolution of problems, either because the user raised them or because other users had previously raised them and they converged with the doubts they had. Other categories were the facilitation of collaboration, easy and direct communication, quick and timely responses, space for debate and discussion, ease of use, and the creation of possibilities for the formation of groups. The students perceived the forums as a space to find help among classmates and also from teachers thanks to their participation in the discussions. The forums also enabled easier and more direct communication with classmates, teaching assistants and teachers in a friendly tone, and therefore simplified the process of resolving doubts. Thanks to the collaboration and effective communication, users reported finding quick and timely answers, without having to wait until class sessions. The students also indicated that the forums allowed them to debate and discuss on assignments, thus seeking to erect forms of validation of the knowledge built individually when solving programming problems.

The second theme was called forum difficulties. This thematic grouped difficulties or problems that the students pointed out when using the forums as a complementary resource to the lectures. The first category shows that students perceived that in their classes there was little use of the forums either because the tool was not integrated in the development of the courses or because of problems at the time of registration or lack of clarity of the forums. The second category grouped comments referring to the attribution of little usefulness of the forums to resolve doubts or obtain help in order to better understand topics about which they were still learning. Thirdly, the students pointed out that an inherent problem of the forums is that their success depends on community use, in other words, the success of the tool depends on the permanent use by the participants, together with the willingness to solve doubts of their classmates in a collaborative spirit. Not all students perceived that there was indeed sufficient participation, and therefore considered that the success of the tool as a collaborative learning resource was highly compromised. Finally, some students found it difficult that the reception of answers through this medium was not immediate. This situation was due to the lack of interaction of classmates in the forums or to the fact that some found it more practical to resolve doubts in synchronous classes or via e-mail.

## DISCUSSION

The results of this study allowed to explore the answers to the main research question in the educational context of the participants: How did the students perceive the contribution of the tools to their learning process in the computer programming course?

With respect to the Notebooks, it can be mentioned, from the data collected through the survey, that most of the students found them to be easy to use. The students considered them as a good tool to learn the topics of the course because of their functionalities that favoured rigorous conceptual learning and because they allowed them to fully understand the topics. Specifically, Notebooks functionalities highlighted by the students were: the possibility of correcting and improving their programs individually or with classmates allowed them a better understanding of the programs; the possibility of documenting the codes they developed made that Notebooks become a source of information to search for precise conceptual definitions; and as a tool to manage theoretical content that could then be easily consulted; and, the possibility that Notebooks gave the teachers to illustrate the concepts by means of applied examples was considered a didactic tool that facilitated the understanding of the topics. On the other hand, from the comments (nine opinions) about some difficulties in using Notebooks, it can be concluded that for some students it is important to carry out training activities before the use of the different functionalities of the tool.

Regarding UNCode, in general, it was well accepted by the students, which is reflected by the level of use of the tool and by the specific benefits that the students found in the tool with respect to their learning process. The use of this tool by the participants in this study was extensive: 100 students made use of UNCode with a total of 19,420 submissions. Additionally, considering that 26.1% of the total submissions were accepted, it can be deduced that the students used UNCode as a tool to test their programs and not only as a means to submit their assignments.

Regarding the usefulness of UNCode for learning, students indicated that the tool was useful for them to better understand the programming problems they were solving and evaluate their solution through test cases or conditions other than those initially proposed, improve their algorithmic thinking skills and learn autonomously. These benefits were due to the fact that the automatic evaluation tool allowed them to test the operation of the programs they designed, to practice their programming skills with the tool outside the classroom, and to identify and correct errors in their programs. The improvements that the students proposed for UNCode seek to expand the current features already offered by the tool to facilitate the students' learning processes.

Finally, with respect to the Piazza forums, the quantitative results allow to conclude that the group of students who had a teaching assistant made greater use of the tool. This result would indicate that it is important to have an expert person in charge of answering the questions of the participants because it motivates the other students to participate in the resolution of the questions and to raise their own concerns. From the quantitative results of the survey, 63 out of 89 students indicated that the forums were useful for learning; however, 16 students did not agree at any level with this usefulness and ten students indicated the NA option (not applicable), which would reflect that they did not know the possibility of using this tool in the course. Although this level of disagreement or lack of knowledge is not a majority, it is a result that reflects some lack of dynamics in use of the tool by some participants in this study that did not have a teaching assistant.

The qualitative data indicate that the students who used the forums found them useful for learning because they were able to resolve doubts about the problems to be solved, improve the understanding of the exercises, find better solutions to the problems and improve programming practices. Additionally, students perceived the forums as a space to find help among classmates and also from teachers thanks to their participation in the discussions, as well as the validation of the knowledge built individually when solving programming problems. The forums made it possible to find quick and timely answers, without having to wait until the class sessions.

It is noteworthy that some of the challenges indicated by Rogalski and Samurçay on the acquisition of programming knowledge and skills, such as the challenges of program understanding, adjustment, debugging (and documenting) [3], were pointed out by the participants in this study as aspects that were favoured by the use of the three tools incorporated in the development of the course included in this study.

## CONCLUSIONS

The results of this study allowed to explore the response, in the educational context of the participants, to the research question posed: How did the students perceive the contribution of the tools to their learning process in the computer programming course? In general, the participants found Notebooks, the automatic evaluation tool (UNCode) and the forums in Piazza easy to use. In this same sense, some students pointed out the convenience of an initial training to better understand how to use Notebooks, and it was found that the forums were used to a greater extent by the group of students who had a teaching assistant. Regarding the perceptions of the usefulness of the tools for learning computer programming topics, it is worth noting that the students found the three tools were useful to better understand the programming problems to be solved. Notebooks allowed students to improve the learning of concepts, to order information that they could later consult to solve problems and to better understand the topics of the course.

The automatic evaluation tool, UNCode, allowed them to improve their programming skills through autonomous practice without the presence of the teacher and to evaluate their solution through test cases or conditions different from those initially proposed. Finally, the forums also allowed them to ask their peers about their doubts about the topics, find better solutions to the problems, validate the knowledge acquired individually with their peers and find quick answers to questions. As future work for this research, the possibility of carrying out a study with quasi-experimental design allows comparing the behaviour of variables, such as academic performance, learning motivation, learning strategies, among others, between students who have these tools available and students who take the course without the support of them. Additionally, it is convenient to make some modifications in the course activities in order to achieve the indicated improvements in the use of Notebooks and forums.

REFERENCES


1. Robins, A., Rountree, J. and Rountree, N., Learning and teaching programming: a review and discussion. *Computer Science Educ*., 13, **2**, 137-172, (2003).
2. Chou, P-N. and Hsiao, H-C., An alternative learning strategy to support engineering students' programming skills: a case study. *Global J. of Engng. Educ*., 13, **1**, 6-11 (2011).
3. Rogalski, J. and Samurçay, R., *Acquisition of Programming Knowledge and Skills*. In: Psychology of Programming. Elsevier, 157-174 (1990).
4. Becker, A. and Quille, K., 50 years of CS1 at SIGCSE. *Proc. 50th ACM Technical Symp. on Computer Science Educ*., 338-344 (2019).
5. Barba, L.A., Barker, L., Blank, D., Brown, J., Downey, A., George, T., Heagy, L., Mandli, K., Moore, J., Lippert, D., Niemeyer, K., Watkins, R., West, R., Wickes, E., Willing, C. and Zingale, M., Teaching and Learning with Jupyter. Jupyter for Education - GitHub: San Francisco, CA, USA (2019).
6. Guo, P.J., Online python tutor. *Proc. 44th ACM Technical Symp. on Computer Science Educ.*, 579 (2013).
7. Restrepo-Calle, F., Ramírez-Echeverry, J.J. and González, F.A., UNCode: interactive system for learning and automatic evaluation of computer programming skills. *EDULEARN18 Proc*., 1, 6888-6898 (2018).
8. Restrepo-Calle, F., Ramírez-Echeverry, J.J. and González, F.A., Using an interactive software tool for the formative and summative evaluation in a computer programming course: an experience report. *Global J. of Engng. Educ*., 22, **3**, 174-185 (2020).
9. Piazza, The Incredibly Easy, Incredibly Engaging Q&A Platform (2021), 24 November 2021, https://piazza.com/


BIOGRAPHIES

Jhon Jairo Ramírez-Echeverry received his Bachelor's degree in electronics engineering from the National University of Colombia (Universidad Nacional de Colombia), Manizales (Caldas), Colombia, the MSc degree in telecommunications engineering from the Universidad Nacional de Colombia, Bogotá, Colombia, and the PhD degree (*cum laude*) in engineering of projects and systems from the Polytechnic University of Catalonia, (Universitat Politècnica de Catalunya) - BarcelonaTech, Spain, in 2017. He is currently an associate professor in the Department of Electrical and Electronics Engineering at the Universidad Nacional de Colombia, Bogotá, Colombia. His research interest areas are engineering education (self-regulated learning) and electronics telecommunications systems.

Felipe Restrepo-Calle received the PhD degree (*cum laude*) from the University of Alicante, Spain, in 2011. He worked as a postdoctoral research fellow at the University of Seville, Spain, in 2012 and 2013. Since 2014, he has been working in the Department of Systems and Industrial Engineering at the National University of Colombia (Universidad Nacional de Colombia), Bogotá, Colombia, where he is an associate professor and leads the Programming Languages and Systems (PLaS) research group. His fields of research interest include programming languages, dependable design in embedded systems and engineering education.

Fabio Augusto González Osorio received his Bachelor's degree in computing systems engineering, the MSc degree in mathematics from the National University of Colombia, Bogotá, Colombia, and the MSc and PhD degrees in computer science from the University of Memphis, United States. He is currently a full professor in the Department of Computing Systems and Industrial Engineering at the National University of Colombia (Universidad Nacional de Colombia), Bogotá, Colombia, where he leads the Machine Learning, Perception and Discovery Laboratory. His research interests revolve around machine learning and its applications in information retrieval, computer vision, natural language understanding, and biomedical image analysis, with a particular focus on the representation, indexing and automatic analysis of multimodal data (data encompassing different types of information - textual, visual, signals).