MEDI@ÇÕES
Revista OnLine

EDUCAÇAO
ESCOLA SUPERIOR
POLITÉCNICO SETÚBAL

A proposal of a Didactic Programming Unit for in-service teachers based on the Bloom Taxonomy to teach block programming languages in the University

DIANA PÉREZ-MARÍN

diana.perez@urjc.es,

Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos

PEDRO PAREDES

pedro.paredes@urjc.es

Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos

Resumen

Este artículo propone el uso de una unidad didáctica basada en la taxonomía de Bloom para enseñar programación a futuros profesores. La hipótesis fue que el uso de la unidad didáctica resultaría en altos niveles de satisfacción además de un mayor rendimiento académico. Un experimento con 70 futuros profesores durante el curso 2022/2023 prueba la hipótesis y abre la investigación de la Didáctica de la Programación no solo para enseñar a los estudiantes de Educación Primaria sino también para formar a los futuros profesores de Educación Primaria.

Palabras clave:

Enseñanza de la programación; Scratch; taxonomía de Bloom; Educación Primaria; profesores en formación; lenguajes de programación por bloques.

Abstract

This paper proposes the use of a didactic unit based on the Bloom Taxonomy to teach programming to pre-service teachers. The hypothesis was that it will result in high satisfaction levels as well as high academic performance. An experiment with

70 pre-service teachers during the 2022/2023 academic year proved the hypothesis. This study opens the research towards a Didactics of Programming needed to teach programming not only in the schools but also for future teachers in Primary Education degrees using active teaching methodologies.

Key concepts:

Programming teaching; Scratch; Bloom's taxonomy; Primary Education; pre-service teachers; block-based programming languages.

Introduction

Teaching programming in Computer Science Degrees is essential (Satorre Cuerda et al. 1996; Cordero et al. 1996) both for structured languages as well as oriented-object programming languages (Fernández Muñoz et al. 2002). Given the characteristics of university students, the objectives at these levels usually include defining and analyzing problems, designing an algorithm to solve it, coding the algorithm in some programming language, and executing the code to validate that the program solves the indicated problem. This can be done using the computer itself with programs that support the teaching of programming (Pérez Calderón, 2008) or without the need to use a computer program with teaching strategies such as collaborative learning (Revelo Sánchez et al. 2018).

In recent years, the advantages of teaching programming from an early age have also been investigated (CSTA, 2012; Jacobsen, 2014; Hromkovic et al. 2019). Children often learn programming with multimedia environments such as Scratch (Resnick et al. 2009), with robots such as Lego WeDo or Mindstorms EV3 (Zygouris et al. 2017), or "unplugged" approaches (without technology) with exercises from sites such as Code.org (Brackmann et al. 2016).

Directly applying the programming teaching scheme from university levels to pre-university levels does not seem the most appropriate due to the different abstraction and processing capacities of children.

In our previous work, we have published a programming teaching methodology for pre-university levels based on the use of metaphors (Pérez-Marín et al. 2020) capable of improving the programming ability of children between 10-12 years old. This work has made us reflect on the importance of teaching future Primary Education teachers also to teach programming. To focus not so much on these teachers learning to program but on learning what could be called Programming Didactics (Cruz-García et al. 2021).

In this article, a Programming Didactic Unit is presented based on the use of Bloom's taxonomy (Anderson et al. 2001) to teach future teachers to teach block programming languages to children between 6-8

years old. The elements of the didactic unit: description, teaching objectives, contents, activities, material resources, organization of space and time, and evaluation are explained for any teacher or researcher who would like to test them into their classrooms.

The teaching objectives are to learn basic programming concepts such as input/output, conditionals and loops, as well as instructions to move objects and create some multimedia backgrounds for their projects. The contents and activities are provided according to the Bloom's Taxonomy to progress from easy instructions to remember to the most difficult task of programming on their own.

The unit is organized into several sessions for one month and it has a final test to check the learning performance of the students. The proposed programming didactic unit was tested with 70 undergraduate students enrolled in the Computer Science subject of the Primary Education Degree during the 2022/2023 academic year. A satisfaction questionnaire was filled in by the students as well as they took a final test registering both high satisfaction and academic performance levels.

1. Framework

1.1. Teaching programming in Primary Education

Teaching programming to children began to be investigated in the 1980s (Papert, 1980). However, this research was paused in the following decades mainly due to the complexity of teaching programming and the lack of trained teachers who could teach it in a subject integrated into the Primary Education school curriculum. It has been taken up again in the last decade with the emergence of new multimedia programs such as Scratch that facilitate the teaching of programming in school environments (Resnick et al. 2009), and worldwide initiatives to integrate the teaching of programming into the school curriculum (Heintz et al. 2016; Hijón-Neira et al. 2017).

One of the most used teaching approaches is constructionism (Papavlasopoulou et al. 2019), recording children's ability to build programs with puzzle pieces that fit together with Scratch. Figure 1 shows a screenshot of a basic program created with Scratch (Resnick et al. 2009).

As can be seen in Figure 1, the program begins by pressing the green flag, and has two instructions, the first for movement that makes the cat move 10 steps and the second for appearance so that it says

"Hello". All the instructions from the different blocks can be combined as a puzzle and execute them on the cat or another object.

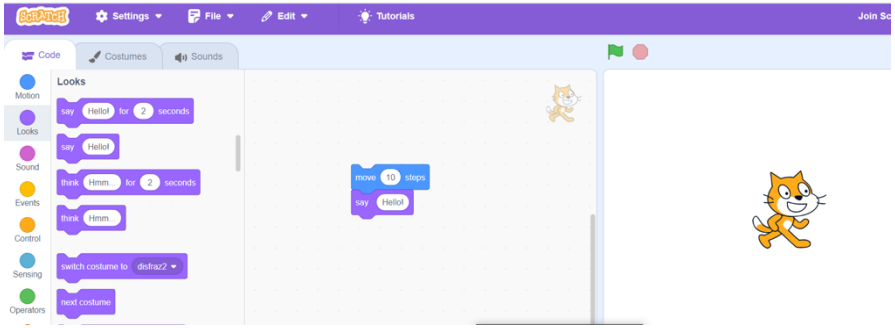

Fig. 1. A screenshot of a simple program in Scratch

Instructions in Scratch are grouped into several categories: motion, appearance, sound, events, control, detection, operators, variables, and others. Each category has a different representative color. Each instruction is contained in a block with the parameters in editable spaces. Children choose a block with the instruction and drag it to the program editor. The blocks fit together like puzzle pieces. The program is executed by double clicking on the program (i.e. the set of blocks) or by clicking on the green flag and the result is displayed on the screen.

Scratch is a very visual multimedia-oriented environment with many possibilities such as changing backgrounds, configuring characters or adding sounds. In this environment, it is impossible for children to make syntax errors since they do not write instructions, they only enter the value of the parameters in the indicated spaces.

Children who use Scratch tend to show very positive attitudes and motivation towards programming (Papavlasopoulou et al. 2019). In any case, approaches based on the use of moving robots such as Lego WeDo or Mindstorms EV3 (Zygouris et al. 2017) have also been tested in the literature. It can be associated with codes that are built in applications on tablets or mobile phones and when executed on, instead of seeing the result of the program on the screen, it is the robot that moves as programmed; and even "unplugged" approaches (without technology) in which students do not program using computers, tablets, mobile phones or robots, but rather do printed exercises from sites like Code.org or games by speaking and writing (Brackmann et al. 2016).

However, the initial problem of the lack of teachers trained to teach programming persists and makes it difficult to introduce the subject of Programming at pre-university levels. In the literature, there are

multiple articles on the Didactics of Programming in university environments (Satorre Cuerda et al. 1996; Cordero et al. 1996; Fernández Muñoz et al. 2002; Kereki, 2017; Pérez Calderón, 2008; Revelo Sánchez et al. 2018) and yet, a shortage of articles related to the didactics of programming for future Primary Education teachers is detected, limiting themselves to providing guides based on metaphors (Pérez-Marín et al. 2020) or the use of gamification with educational video games (Cruz-García et al. 2021). Therefore, it is necessary to continue researching the proposal of a Programming Didactic for future Primary Education teachers.

1.2. Bloom's taxonomy

Bloom's Taxonomy is a list of objectives (or levels) that evaluate the learning process of any student, as well as a useful starting point to logically design activities and exercises and achieve meaningful learning that lasts throughout life. Created in the 1950s by Benjamin Bloom, psychologist and pedagogue at the University of Chicago, it is based on a hierarchy of educational objectives that are sought to be achieved with students, dividing them into three areas: cognitive, affective and psychomotor. From the cognitive field arises the pyramid of Taxonomy, which consists of six categories with different 'verbs' (actions that can be performed at each level). These help to evaluate following an evolution from lower to higher complexity depending on the cognitive process required by a specific task.

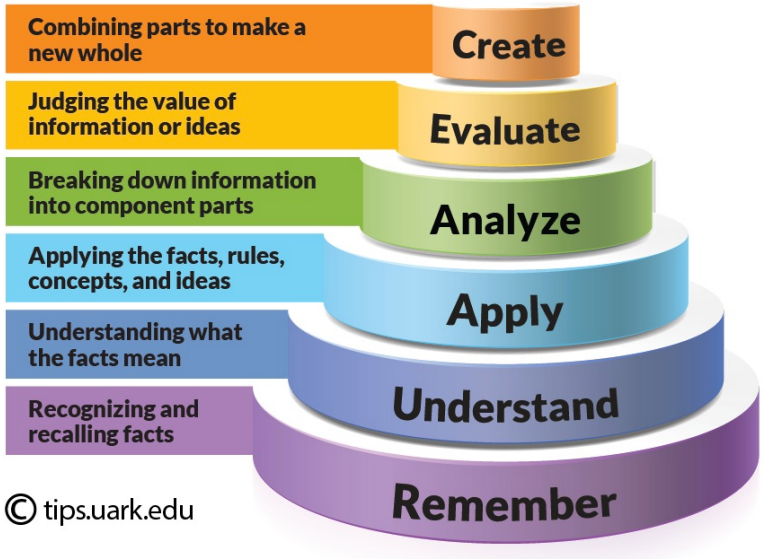

Fig. 2. Bloom's taxonomy (Anderson et al. 2001)

The categories of Bloom's Taxonomy have gone through different changes to adapt to the digital age and, currently, one of the most accepted updates is that of researchers Anderson and Krathwohl (see

Figure 2) which is made up of the following levels:

- **Remember:** It is the basis from which all learning is based and refers to the ability to remember specific facts, methods, processes, schemes or frames of reference in the long term.
- **Understanding:** It is the student's ability to know what is being communicated and requires an abstract thinking capacity. It is about knowing how to interpret information and being able to express it in your own words.
- **Apply:** It consists of putting into practice the concepts and procedures seen previously. It involves using the elements studied in other situations.
- **Review:** It is based on breaking down a problem into parts, considering them separately and discovering the relationships between them to, finally, draw conclusions.
- **Evaluate:** It is related to the issuance of value judgments (quantitative and qualitative) regarding the information and methodologies received.
- **Create:** This category was included by the authors and is the most complex: it is based on using what has been learned to build and develop new ideas or on proposing solutions to day-to-day problems.

Regarding the application of Bloom's taxonomy to the teaching of computer science, there are several studies that have attempted to apply it to evaluate students, but none have done so to establish a sequence of activities adapted to the levels of the model (Masapanta-Carrión & Velázquez-Iturbide, 2017).

2. Programming Didactic Unit

In this section, the following elements of the didactic unit are described and proposed for any researchers or teachers who would like to follow them for their research/teaching: description, teaching objectives, contents, activities, material resources, organization of space and time, and evaluation. It is important to investigate into these elements because up to date there is not a clear consensus in the literature neither on the contents, methodologies, or activities most adequate to teach programming to children and future teachers that have to teach programming to children, despite all the benefits investigated of learning programming at early stages.

Description

The goal of the didactic programming unit is to teach children block

programming languages such as Scratch. The target children are not expected to have previous knowledge on Scratch as they are between 6-8 years old.

Teaching objectives

1. Students will learn the instructions of the Scratch environment.
2. Students will know how to use the Scratch environment.
3. Students will understand basic concepts of programming such as variable, input and output, conditional, loop and their application to programming.
4. Students will apply the Scratch instructions to program the concepts learnt and also to move objects and create some multimedia backgrounds for their projects.
5. Students will Review and evaluate Scratch programs.
6. Students will combine programming concepts to create a block-based program on their own using Scratch.

Table 1. Bloom's taxonomy to teach programming

Level	Content
Remember	The instructions of the Scratch, categories and Scratch's interface
Understand	Basic programming concepts such as variable, input and output, conditional and loop.
Apply	Programming exercises from easy to more difficulty
Review	Block-language programs such as Scratch
Evaluate	Scratch programs with errors
Create	Goals of the programs to be created by the students

Contents, activities and material resources

Table 1 shows a summary of the contents according to the levels of the Bloom's taxonomy. This taxonomy has been chosen because it helps to organize the activities according to their level of difficulty. Moreover, it can be combined with other methodologies such as Pro-

ject-Based Learning because all the activities proposed are to be included in a long-term project that students are working during the subject. The Bloom's taxonomy is the guide to work during the project.

For the first level of Remember, it is proposed to begin teaching games so that Primary Education students learn the Scratch interface to the students. Initially, the interface would be taught to university students so that they can visualize the possibilities of Scratch and where to find each instruction according to its type.

Later, teachers can propose students to play games with the different colors of the instructions to liven up the teaching of Primary children. For example, dividing the class into colored teams, and each team being responsible for reciting instructions to the rest. For instance, the blue team can start reciting the sequence instructions to the rest of the class, then the yellow team reciting the event instructions, and so on until the entire class remembers the programming instructions.

For the second and third level of Understand and Apply, it is proposed to provide students with a list of exercises, classified by programming concepts that they have to understand, such as input/output, conditionals and loops. University students must first be able to understand how to solve the exercises and then share in class how this same list could be solved by Primary Education students. Some examples of exercises to understand the concept of conditional are the following:

- Write a program to find out the greater of two numbers.
- Write a program so that if you fail a test, the cat tells you to study more, and if you pass the test, the cat congratulates you.

For the fourth and fifth level of Review and Evaluate, it is proposed to provide programs made to university students who must be able to understand, compare, and detect if they have errors to solve them first, and thus empathize with the future situation in which they will be.

Finally, for the **Create** level, which is the one that requires higher-order thinking, university students should be given the choice of a topic to teach their future Primary Education students using Scratch and they should be the ones to decide what. create a program to achieve the goal they set for themselves. This will allow them in their future classes to also transmit to their students the idea that Scratch and programming in general is not limited only to solving exercises that teachers give them, but ultimately the objective is that in the face of any problem that they themselves face, suggest they can solve it by creating a new program. In addition, university students are asked to

record a group video explaining programming concepts to also evaluate their oral expression ability.

Organization of space and time

Each level should be taught at least for one week (4 hours, in 2 sessions of 2 hours) in a classroom with computers or tablets, and chairs that can be moved to make collaboration between students easier. Lower levels can be worked in big group with the teacher and all the class, while higher levels can be worked in groups of 4-5 students, pairs and even provide some individual tasks at the higher creation levels to favor the individuality of each student.

Each session should start with a review of the preview lesson to refresh the concepts and activities seen. It is possible that some students advance faster than others. Moreover, to attend the diversity, several rhythms, learning styles, needs and preferences should be taken into consideration by creating zones inside the classroom so that:

- Students struggling with lower levels of the Bloom's taxonomy can be in a *working zone* with constant support from the teacher.
- Students in the upper levels of the Bloom's taxonomy can be given more freedom in a *playground zone* with their groups creating new programs and testing new ideas without so much guidance by the teacher.

Evaluation

To test whether the students have been able to achieve the teaching goals and cover the contents, the evaluation should be continuous during all the sessions. Moreover, a formative assessment with feedback per activity should be provided so that students can identify their failures to avoid making them again.

An individual final test is also proposed to check whether the students have assimilated the concepts and are able both to Review programs and create programs with Scratch on their own.

A sample question for this final test to check whether the Review level has been reached could be the one shown in Figure 3 with this statement:

The output for this program would be:

It does not work

Menor

Mayor

None of the previous answers

That way, students have to execute in their mind (without using Scratch) the program and provide as answer the output of the program if they have understood the conditional concept.

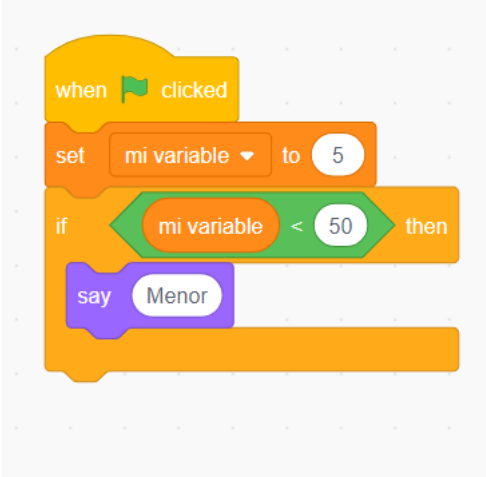

Fig. 3. Sample question for the exam

Finally, it is also possible to add open questions asking students to create programs to combine the concepts under study and execute them in Scratch to see if they are correct. For instance, students could be asked to write a program to calculate basic operations.

4. Experiment

A. Sample

The programming didactic unit based on the Bloom's taxonomy was applied during the 2022/2023 academic year in the first year of the Primary Education Degree in the subject of Computer Science and Teaching Digital Competence to the 70 students enrolled in the subject. However, when they were asked to fill out a form with their opinion anonymously and voluntarily, only 25 completed it, which will be the sample with which we will work for the data collected.

56% of the students are between 18-19 years old (except for 4 students, 16%, who are between 17-18 years old and 7 students, 28%, who are over 19 years old). 64% of the students are men compared to 36% who are women. 76% of the students already knew Scratch from their previous training.

B. Procedure

In January 2022, the subject of Computer Science and Digital Teaching Competence began, which is from the second quarter and lasts until May 2022. Following the method described in the third section,

the month of April and a week in May were dedicated (since in between it was Easter and there were some non-school days).

At the end of the subject, the students were asked to complete a questionnaire anonymously and voluntarily to find out the degree of satisfaction with the way in which they had learned to teach programming in Primary Education. It was decided to use an ad hoc questionnaire since no other validated option was found that covered the data questions that needed to be collected to know the degree of student satisfaction with the new teaching method applied during the course.

The questionnaire created consisted of 15 questions, the first three to record their age, sex and whether they knew Scratch previously and to know the study sample; the following 11 multiple choice questions with the possibility of choosing only one answer to find out your opinion on what you thought of the programs carried out, and your degree of satisfaction.

To encourage students to complete the entire questionnaire, a task that they usually do not like and due to the voluntary nature of the activity, it was decided to use star scales in some questions to avoid all questions having traditional Likert type answers, true/false, yes/no and provide variety. The last question was left open and without obligation to complete solely to collect any other aspects that the students might freely want to add.

C. Results

4.1 in a scale from 0 (worst) to 5 (best) with a standard deviation of 0.88. 64% of students consider that learning to teach programming in this way has been satisfactory and 24% consider that it has been very satisfactory. 88% also indicate that they liked learning to teach programming with Scratch.

This high level of satisfaction may be since the students have found all the levels of Bloom's taxonomy covered. When the students are asked what they consider having been the most difficult thing about learning to teach programming, the majority – 68% – indicate that it was the sequence of instructions.

Finally, when free comments are collected that students voluntarily want to contribute, they provide responses of this style:

- "I found the Scratch part very useful to introduce programming to children in this technological era."
- "I consider that learning Scratch has served as an example for us to carry out activities with the students."
- "I liked the way in which Scratch has been used, as a general

vision to be able to use that tool in our future as teachers."

Regarding the academic performance of the students, it was remarkable with a 7.7 average score on a scale that goes from 0 to 10.

Conclusions

The teaching of programming in Primary Education has generated great interest in recent decades. In this article, the focus has been on how to teach programming to future teachers in Primary Education Degrees. The use of Bloom's taxonomy has been proposed, covering from the most basic levels of remembering to creating, in an experience with the students of Computer Science and Digital Teaching Competence of the Primary Education Degree of the Rey Juan Carlos University in the 2022/2023 academic year.

The results obtained seem to indicate that this didactic approach could be beneficial for students since it has obtained high levels of satisfaction with a 4.1 value on a scale of 1 to 5 stars; and, 88% of students indicating that they found it to be a satisfactory or very satisfactory teaching experience and indicate that they liked learning to teach Scratch to their future students in this way.

Furthermore, 100% of the students obtained a grade higher than 5 in the exam, and the average was 7.7. It should also be noted that no student dropped out of the subject, this may be due to the methodology used and the high degree of student satisfaction with it.

As future work, we want to continue researching the most appropriate teaching methodologies to teach programming to future teachers and the scientific community is encouraged to contribute more work in this line.

Acknowledgmens

Research work carried out within the framework of projects P2018/TCS-4307, which has also been co-financed by the ESF and ERDF structural funds, and Grant PID2022-137849OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by ERDF, EU.

References

Anderson, L.W., Krathwohl, D.R., Airasian, P.W., Cruikshank, K.A., Mayer, R.E., Pintrich, P.R., Raths, R., Wittrock, M.C. (2001). "Taxonomy for Learning, Teaching and Assessing. A Revision of Bloom's Taxonomy of Educational Objectives", Addison-Wesley Longman: Nueva York, NY

Brackmann, C., Barone, D., Casali, A., Boucinha, R., Muñoz-Hernandez, S. (2016) "Computational thinking: Panorama of the Americas". Presented at Computers in Education (SIIE), pp. 1-6, IEEE.

Computer Science Teachers Association (CSTA, 2012). "Computer Science K–8 Building a Strong Foundation", http://csta.acm.org/Curriculum/sub/CurrFiles/CS_K8_Building_a_Foundation.pdf.

Cordero, J.M., González, R., Romero, R. Martínez. (1996). "Introducción a la programación, un enfoque práctico", Algaida.

Cruz-García, I., Martín-García, J.A., Pérez-Marin, D., Pizarro, C. (2021) "Propuesta de didáctica de la Programación en Educación Primaria basada en la gamificación usando videojuegos educativos". Education in the Knowledge Society (EKS), 22, e26130-e26130

Fernández Muñoz, L., Peña, R., Nava, F., Velázquez Iturbide, A. (2002). "Análisis de las propuestas de la enseñanza de la programación orientada a objetos en los primeros cursos". VIII Jornadas de Enseñanza Universitaria de la Informática, 433-440.

Heintz, F., Mannila, L., Färnqvist, T. (2016). "A review of models for introducing computational thinking, computer science and computing in K-12 education", IEEE Frontiers in Education Conference, pp. 1-9.

Hijón-Neira, R., Santacruz-Valencia, L., Pérez-Marín, D., Gómez-Gómez, M. (2017). "An analysis of the current situation of teaching programming in Primary Education". In 2017 International Symposium on Computers in Education (SIIE) (pp. 1-6). IEEE

Hromkovič, J., Komm, D., Lacher, R., Staub, J. (2019). "Teaching with LOGO Philosophy". Encyclopedia of Education and Information Technologies, A. Tatnall (ed.), Springer Nature, https://doi.org/10.1007/978-3-319-60013-0_76-1

Jacobsen, H. (2014). "Five-years-old learn coding in schools to prepare for future labour market". EurActiv.com - EU News & policy debates, across languages.

Kereki, I. (2017). "Enseñando y Aprendiendo Programación Orientada a Objetos en los primeros cursos de Programación: la experiencia en la Universidad ORT Uruguay". Universidad ORT Uruguay, Uruguay. https://silo.tips/download/enseando-y-aprendiendo-programacion-orientada-a-objetos-en-los-primeros-cursos-d

Masapanta-Carrión, S., Velázquez-Iturbide, J.A. (2017). "Una Revisión Sistemática del Uso de la Taxonomía de Bloom en la Enseñanza de la Informática", SIIE 2017, pp. 294-299

Papavlasopoulou, S., Giannakos, M.N., Jaccheri, L. (2019). "Exploring children's learning experience in constructionism-based coding activities through design-based research", Computers in Human Behaviour, DOI: 10.1016/j.chb.2019.01.008

Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York, NY: Basic Books

Pérez Calderón, R. (2008). "Una herramienta y técnica para la enseñanza de la programación". CICos, 229-239

Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., Pizarro, C. (2020). "Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children?". Computers in Human Behavior, 105, 105849.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K. (2009). "Scratch: Programming for all". Communications of ACM, 12, 52(11), pp. 60-67

Revelo Sánchez, O., Collazos Ordoñez, C.A., Jiménez Toledo, J.A. (2018). "El trabajo colaborativo como estrategia didáctica para la enseñanza/aprendizaje de la programación: una revisión sistemática de literatura". TecnoLógicas, vol. 21, no. 41, pp. 115134

Satorre Cuerda, R., Llorens Largo, F., Puchol García, J.A. (1996). "Enseñar Programación en las Ingenierías Informáticas". II Jornadas Nacionales de Innovación en las Enseñanzas de las Ingenierías, Instituto de Ciencias de la Educación, Universidad Politécnica de Madrid, Comunicaciones Volumen II, pág. 840-847.

Zygouris, N., Striftou, A., Dadaliaris, A., Stamoulis, G., Xenakis, A., Vavougios, D. (2017). "The use of LEGO Mindstorms in elementary schools". EDUCON, 514-516, doi:10.1109/EDUCON.2017.7942895

Nota curricular

Diana Pérez-Marín es Profesora Titular de la Universidad Rey Juan Carlos, Madrid, España desde el año 2018. Recibí el Doctorado Europeo en 2007 en Ingeniería Informática y Telecomunicación, y me gradué en Ingeniería Informática en 2002 por la Universidad Autónoma de Madrid. Trabajé como profesora e investigadora en la Universidad Autónoma de Madrid durante 10 años, y soy profesora e investigadora en la Universidad Rey Juan Carlos desde el año 2009. Soy miembro del Laboratorio de Tecnologías de la Información en Educación (LITE). Mis áreas de investigación son la Interacción Persona-Ordenador y la Educación Asistida por Ordenador. Campos en los que he publicado más de 100 artículos en revistas y conferencias nacionales e internacionales.

Pedro Paredes es Maestro en Educación Infantil, Licenciado en Lingüística y Doctor en Ingeniería Informática y Telecomunicación por la Universidad Autónoma de Madrid (UAM). Desde enero de 2019 soy profesor en la Escuela Técnica Superior de Ingeniería Informática (ETSII) de la Universidad Rey Juan Carlos (URJC). Llevo desde 2001 impartiendo docencia universitaria e investigando. Tengo más de 20 publicaciones en revistas y congresos internacionales y he participado en 5 proyectos nacionales, otro a nivel de Comunidad Autónoma, otro para jóvenes investigadores de la UAM y otro en colaboración con el Ayuntamiento de Fuenlabrada (INGÉNIATE). He tenido el placer de trabajar en distintas universidades públicas y privadas. Mis intereses han sido siempre la aplicación de tecnologías para la mejora del aprendizaje: integración de estilos de aprendizaje en sistemas hipermedia adaptativos para la educación, aplicaciones para personas con discapacidad cognitiva, juegos para enseñar a los niños a cómo reaccionar ante situaciones de emergencia, el uso de laboratorios remotos para la enseñanza y, en la actualidad, la enseñanza de la programación por bloques en niveles preuniversitarios. En mi metodología docente intento aplicar los descubrimientos y avances en mis investigaciones y las de muchos otros: clase invertida, identificación y aprovechamiento de los estilos de aprendizaje de mis alumnos o adaptación a las distintas capacidades.