Teaching C/C ++ programming using a multimedia system with videos

José Galindo[1], Patricia Galindo[1], José María Rodríguez Corral[2],

[1] Dpto. Lenguajes y Ciencias de la Computación, University of Málaga, Spain.
[2] Dpto. Dpto. Ingeniería Informática, University of Cádiz, Spain.

Abstract. The C programming language is widely used in computer and industrial engineering. Because of that, such programming language is also widely used as a language to teach programming to industrial engineering students. In Spain, many universities use this language compulsory in the first year, or even in higher courses. Our experience shows that learning computer programming in four months is an arduous task, but curricula require it. Such learning process is also tough by the fact that many students cannot attend classes regularly and, even if they attend, sometimes the class is no longer at the level they require. In this work we develop a series of files in "presentation" format (.ppsx) and videos that allow students to see several explanations about the most complicated programming C topics: functions, arrays, structures, strings, arrays with structures, etc. This multimedia material includes explanations (voice-over), and animations with examples. Students can watch and listen to the explanations whenever and wherever they want (tablet, PC, phone…). Surveys made to students reveal that it is also interesting for students who regularly attend classes, and they prefer to use this course material only at home, outside of regular classes.

Keywords: programming languages, multimedia teaching, autonomous learning, C/C++ language.

1 Introduction

It has been noted that students frequently experiment difficulties when learning traditional programming languages (such as C, C++ and Java) and abstract programming concepts [5][7][10].

In order to make the creation of new applications easier for people without programming skills, recent software developments include programming tools that hide much of the complexity of traditional languages. For example, a Visual Programming Language [3] could simplify many programming tasks being suitable also for learning purposes even in specific contexts, such as robot programming [13]. On the other hand, authoring tools allow a person to develop an application simply by linking together objects, such as a picture, a sound or a text paragraph. Authors can create useful and attractive graphics applications by simply describing the object's relationship among themselves and by sequencing them in an appropriate order. This

could be suitable for novice programmers [6], including those students in the first years whose university degrees are not directly related to the computer science discipline.

Basic programming concepts are easier to learn in a visual programming languages, but commercial and industrial applications are usually made using traditional languages. On the other hand, some kind of visual languages, such as Visual Basic or Visual C#, need to program with traditional language techniques in order to be efficiently and accurately used. The relevance of traditional languages is proved by the fact that these languages are very common in the universities, in many degrees such as computer sciences, telecommunications engineering, industrial engineering and so on. In fact, the C language is used, along with the assembly language, for programming microcontrollers, and the Arduino language [11] is based on C++. Likewise, Matlab (MATrix LABoratory) is a software very used in the field of scientific computing in universities, research institutes and industries around the world. It can be used for representation of graphics, mathematical computation, modeling and simulation, data analysis and processing and development of algorithms.

Particularly, in Spain there are a lot of subjects in first courses of different degrees in which their first goal is to learn to program using a traditional programming language (such as C, C++, java, Python, Visual C#, Matlab…). In addition, the majority of these subjects are taught in 15 weeks, at a rate of about four and a half hours of class per week. In this context, the students have to face the following main problems [4]:

- The high level of demand, especially in technical degrees, which are the majority.
- The scarce time to learn the theoretical concepts and mainly, to acquire the ability to put them into practice while programming with certain complexity: The exams include the most intricate problems seen in class, but only one or two weeks are devoted to such kind of problems.
- Numerous repeating students who do not attend all classes due to incompatibility of their schedules.
- Little practical teaching time to practice: Normally, students receive, each week, 3 hours of theoretical classes (on the blackboard) and 1.5 hours of practical classes. It is notorious that students must practice more outside university hours and teachers warn them to do so by providing some proposed exercise relationships (with some of them solved).
- Shortage of time to study and practice important concepts and exam exercises. The period of classes is followed by only 1 or 2 weeks before the exam.
- Students who already have certain knowledge have to wait until the rest of their classmates are at the same level, in order to advance together.
- Losing a class might mean being left behind or having to invest a lot of time and effort in catching up.
- To learn computer programming, students need to grasp valid basic concepts during their introductory classes as these forms a strong background for more advanced programming exercises. The more doubts the student has in previous concepts, the more effort he will have to invest in order to advance.

- Besides, a negative perception of the subject is added in many degrees excluding computer science. As beginners, the students believe that learning programming is cumbersome [14].

Moreover, teachers also have to face some problems although, of course, these inconveniences depend on different factors and the teaching context:

- High number of students per group, which prevents giving all classes in a practical way.
- Heterogeneous groups, where some start from scratch but others already have a good level. This aspect, together with the previous point, prevents giving proper treatment to each student according to his or her level.
- Need to repeat the same class over and over again: This may be because the same teacher educates several groups, but it is also referred to repetitions from one quarter or year to the following.
- The teaching quality might vary depending on personal factors of the teacher such as fatigue, noise, multiple repetitions, and distractors, among others.

Of course, it is not easy to solve all these problems for teachers and students in a comfortable and simple way. Some teachers have opted to record their classes on video and uploaded them on the Internet so that their students have access to them 24 hours. Only this solution already solves some of the problems, although it poses other challenges, such as the appearance of the blackboard, which should look good, or the quality of the video (regarding environmental noise, pronunciation…). All these problems are solved by professionally prepared videos or, as we have opted in our case, by preparing different classes in Microsoft Power Point format files (.ppsx files). Among other advantages, the quality of image and sound is excellent. Furthermore, the student has completely control over the explanations rhythm.

A study indicated that the native digital students are drawn to visual media and creative tasks [12]. Therefore, the integration of multimedia as a reflection tool in learning is crucial in order to maintain students' motivation and engagement in the programming class.

In this paper, we present our conclusions after making several C programming chapters in multimedia format for the subject 'Fundamentals of Computer Science' for Industrial Engineering students at the University of Malaga (Spain). Our main conclusion is that students perceive that the multimedia approach as motivating and engaging. In section 2, the methodology used is presented. Subsequently, in section 3, the results of the surveys carried out on the students are commented. In section 4, the advantages and disadvantages of our proposal are discussed and, finally, the document is ended with some conclusions and future lines.

2 Methodology

Learning to program is intensely meticulous for novice undergraduates who do not have any background on computer programming [9]. At the moment, current teaching about theoretical concepts is based on 'chalk and talk' method and textbooks and this does not always work well. Many teachers have substituted the chalk with static

presentations but, in essence, the methodology is very similar to those classes taught thousands of years ago. Of course, practical classes are very different.

Chansilp and Oliver [2] suggest that the graphic representations of algorithms used in most textbooks are abstract visualizations and are not sufficient for learners to develop the logical thinking required in programming courses. The students usually only depend on the lectures and the textbook to develop their programming comprehension. However, just as Annamalai and Salam [1] said, the lecturers only go through the lesson once. Hence, the students have no lesson repetition, while this is crucial to learn and understand better. Students' problems with learning C programming are mostly based on a lack of understanding of conceptual and mental models [8].

Our proposal is based on providing the student with a set of files with the main explanations of several programming parts (such as control statements, inputs and outputs, functions, arrays, strings, structures, searching and sorting algorithms…). The general characteristics of this system are:

- Pace of learning: the student can stop and restart the explanations whenever he wants, as well as repeat them as many times as he wishes.

- Animations and sound: There are animations and sounds that make the experience much more attractive. This, joined with the teacher's voice-over, tries to focus the student's attention. For example, in a visually attractive way the student sees how the value of a variable changes in different statements, as the program is running.

- Step by step explanations: Theoretical and practical concepts are linked together and are explained with examples whose complexity is growing. For example, Fig. 1 shows a program that uses a simple array on the left, and the behavior of the program is explained to the right with animated graphics and tables.

- Sample programs: Complete programs, fragments or example functions are included, explaining their execution step by step. For example, Fig. 2 shows a frame of the animation explaining the selection sort algorithm. Step by step it is explained how the elements of the array are exchanged among themselves as shown by the arrows. It can be seen in the bottom part how at the end of the algorithm the array is ordered. At the end of each topic, various programs with fewer explanations are provided, inviting the student to check, understand and modify them.

- Complex concepts: The most complex concepts are explained in several ways because we have detected that different students understand better the ideas using different techniques for each of them. For example, in Fig. 3, an explanation about passing parameters by reference is shown (using the C++ style), which is an issue that is often extremely difficult to understand by beginner students (even more using the traditional C style).

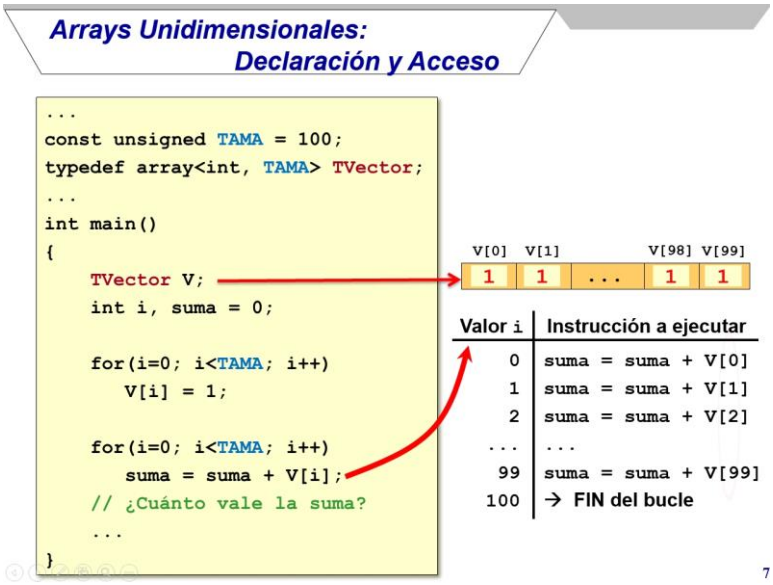

Fig. 1. Example of slide explaining a simple array. Source: own construction.

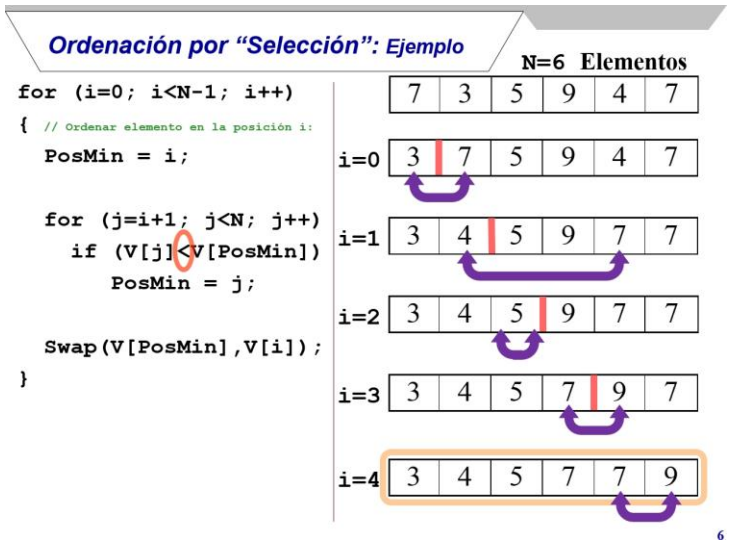

Fig. 2. Example of slide explaining the selection sort algorithm. The comment in green says: `Sort element on i position`. Source: own construction.

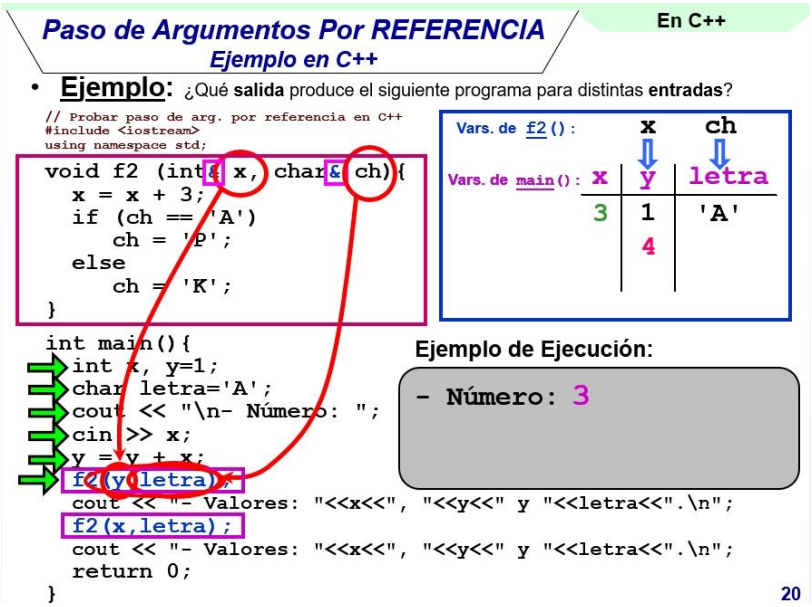

Fig. 3. Example of slide explaining the pass of parameters by reference using C++ style. Source: own construction.

3 Surveys

In order to inform the students about this multimedia material developed, it has been used in one class per course, for several years. After that class, a survey has been passed and they are advised that all the material is uploaded and available via web. In addition, students are asked to fill in an anonymous and optional survey each time they use the material (using the platform www.survio.com). The students are informed that their opinion is extremely important in order to improve the material and to broaden the topics covered.

The most interesting questions and results of these surveys are as follows:

1. Do you find this multimedia format interesting and useful to learn?
   - Average response: 4.6 of 5 (standard deviation: 0.3).
2. Do you think the explanations are adequate?
   - Average response: 4 of 5 (standard deviation: 0.6).
3. Do you think the examples used are appropriate?
   - Average response: 4.2 of 5 (standard deviation: 0.2).
4. Have you repeated or will you repeat all or some parts of the explanations?
   - Three answers: No (0%); All (16.7%); Some parts (83.7%).
5. Do you think it is appropriate to use this material in class or it is better to leave it only for individual study? Five answers:

a) I prefer only the teacher's explanations in class (33.3%);
b) In class, teacher explanations should be mixed with little use of multimedia material (50%);
c) The ideal option is to combine both methods equally (16.7%);
d) The ideal option is to use mainly the multimedia material, and the teacher would only answer questions or add some comments (0%);
e) I prefer only multimedia material in class (0%);

6. Evaluate the speed of the explanations, in general:
   - Three answers: Too fast (16.7%); Good rhythm (83.3%); Too slow (0%).
7. Would you like to have many topics of all your subjects in this format?
   - Average response: 4.3 of 5 (standard deviation: 0.3).

The surveys basically tell us that students appreciate this material, that they have used it and that they will continue using it to study. They also reflect that they do not like that these material are used in class, relegating the teacher's task to solve doubts. In class, students prefer the immediacy and spontaneity of a teacher and the ability to raise their hands and ask at any time a specific question that may arise. That is something that, until now, is not easy to incorporate into an automatic education system.

One of the most important aspects in the development of this type of material is the speed of the explanations. Surveys say that, although most students value the pace of explanations positively, there is a minority who notice it too quickly, and nobody values it as too slow. In our case, we must assess whether to make the explanations a little slower. But this must be done carefully, first because they could become excessively slow and boring and secondly because we must take into account that the student can repeat the explanation whenever he wants. At any time the student can stop the explanation and go back, which facilitates the understanding for those students who notice that the explanations go faster than they would like.

4 Advantages and disadvantages of our approach

In this section, the strengths and limitations of this approach are considered. It is highly difficult to solve simply and quickly all the problems listed in the introduction. However, the material we propose resolves or reduces these problems because of the following aspects:

- It enables the students to learn at their own pace. Students can repeat the explanations whenever they want, wherever they like and as much as they need. This means that if a student misses a class, he or she does not have to be burdened by that. Besides, students can even go ahead of the teacher's explanations, organizing better their available time and fitting it to their schedules. This is especially interesting in repeating students or students with previous knowledge, who do not have to miss the first weeks until their classmates reach their level.

- Students can solve their doubts going through this material and listening to the exact part or lesson that interests them.

On the other hand, teachers have noticed that students who use this material advance better and faster, with less doubt and mistakes.

The available formats (presentation and video) allow students to choose the format they prefer:

- The advantage of presentation files is that presentations are automatically stopped at certain points. This invites the student to take a break to reflect or read something, and move on whenever he or she wants. The drawback is that you cannot go back to a specific point to repeat just from the desired time but you can go back (or advance) by blocks or pages.
- One of the advantages of videos is that the student can go back the video or advance to the exact second from which he or she wants to restart the explanation. Another additional advantage is that videos can be viewed online, from any device, without the need to download it.

As the student survey shows, they prefer not to use this material in class or use it in a timely manner. This is because they prefer to interact with the teacher and the possibility of asking questions just when they arise, something that does not allow the system that we present here, but that could be combined with non-contact tutoring schedules, both offline tutoring (using email or other messaging applications), and online tutoring (using some chat or video calls).

Non-contact tutoring is essential in case of distance teaching or in the case of students who cannot attend classes or tutoring for any reason (incompatibility of schedules, illness ...).

In addition, these tutoring schedules can be individual or in groups (using forums or group video calls). One advantage of group interactions is that they can occur without the direct participation of the faculty, who can play a supervisory role and resolve doubts when no one else does. In this case, the students can help each other solving doubts. The great advantage of this is that a climate of group work and participation enhances learning. For some reason, there are some groups that like this kind of interaction between themselves and other groups in which members do not participate or do very little. However, the teacher's role in favoring that participation can be decisive. In any case, the multimedia material favors the interaction between students: students use it in groups, they recommend others a certain part where something concrete is explained, etc.

To capture the student's attention, different techniques have been used. For example, put an error intentionally, to surprise the student. There are other techniques that we still need to explore, such as asking the student to find an error, telling an anecdote related to the exercise that is being solved, or even a joke related to computer programming.

5 Conclusions and future lines

The introductory programming subjects usually suppose that students do not have previous concepts, but in only 15 weeks, they have to reach a quite high level of

programming (exercises of arrays, structures, nested datatypes, nested loops…). According to our experience, students find difficult to reach an adequate level in such a short time. It would be desirable for these subjects to be annual because, even if the syllabus and the class hours were not extended, the fact of having more time to mature the concepts and practice would make these subjects easier to learn. However, the current structure of the curricula in Spain has led to the subjects to be quarterly instead of annual, which makes our suggestion difficult to be taken into account.

Given this fact, any material or idea that improves learning is well received. In this paper, a multimedia material to learn to program in C language is exposed. It solves or reduces the problems described in the introduction. On the other hand, according to the surveys made to students, this material is well accepted. Although it can be improved, students applaud and use it. It must be highlighted the last question of our survey, in which the students reflect their desire to have this multimedia files for all their subjects.

In the future, it might not exist face-to-face classes and students might have all classes in electronic format (presentations, videos, software, exercises, games…), so that only a few practical classes and classes to solve doubts would be given. Even these classes could also be taught by chat or videoconference. However, that is not our goal when presenting this multimedia material. We think that the relationship between teacher and student is overriding and universities should not ignore the figure of the teacher and much less replace him with electronic means. Nonetheless, this system can reduce the need of teachers and, above all, make it easier for students to access explanations when they wish and not when the schedule stipulates they have to.

As future lines of expansion of this work, it isproposed to introduce more interactivity between the multimedia documents and students. Moreover, it issuggested to translate these files into English to reach a wider audience through the Internet and for exchange students.

Acknowledgments. The authors would like to express their gratitude to the University of Málaga, supporting this work under the PIE17-175 project entitled “Autoaprendizaje de programación de ordenadores”.

References

1. S. Annamalai, & S.N.A. Salam. A Multimedia Approach towards Learning C Programming: A Discussion from Novice Learners' Perspective. Journal of Telecommunication, Electronic and Computer Engineering, Vol. 9 No. 2-12 (2017)
2. K. Chansilp & R. Oliver. Using multimedia to develop students' programming concepts. Proc. of EDU-COM 2002, 91--101 (2003)
3. M. Erwig, K. Smeltzer, & X. Wang. What is a visual language? J. Vis. Lang., Comput., vol. 38, pp. 9--17, (2017)
4. J. Galindo and P. Galindo. Teaching computer programming for industrial engineering without teacher. 7th Teaching & Education Conference (2019)
5. A. Gomes & A. J. Mendes. Learning to program-Difficulties and solutions. Proc. Int. Conf. Eng. Educ. (ICEE), Coimbra, Portugal, (2007)

6. M. S. Horn, E. T. Solovey, R. J. Crouser, & R. J. Jacob. Comparing the use of tangible and graphical programming languages for informal science education. Proc. 27th Conf. Hum. Factors Comput. Syst. (CHI), 975—984. Boston, MA, USA, (2009)
7. E. Lahtinen, K. Ala-Mutka, & H. Järvinen. A study of the difficulties of novice programmers. Proc. 10th Annu. Conf. Innov. Technol. Comput. Sci. Educ. (ITiCSE), Caparica, Portugal, 14--18 (2005)
8. M.Y. Law, C.S. Lee, & Y.T. Yu. Learning motivation in e-learning facilitated computer programming courses. Computers & Education, Vol. 55, no. 1, 218--228 (2010)
9. M. J. Lee & A. J. Koo. Personifying programming tool feedback improves novice programmers' learning. Proc. of the seventh international workshop on Computing education research, 109—116 (2011)
10. I. Milne & G. Rowe. Difficulties in learning and teaching programming-Views of students and tutors. Educ. Inf. Technol., vol. 7, no. 1, 55--66 (2002)
11. S. Monk. Programming Arduino: Getting Started with Sketches. McGraw Hill (2012)
12. S. Naz, S.H. Shirazi, T. Iqbal, D. Irfan, M. Junaid & Y. Naseer. Learning Programming through Multimedia and Dry-run. Research Journal of Applied Sciences, Engineering and Technology, Vol. 7, no.21, 4455--4463 (2014)
13. J.M. Rodríguez Corral, I. Ruíz-Rube, A. Civit Balcells, J.M. Mota-Macías, A. Morgado-Estévez & J.M. Dodero. Study on the Suitability of Visual Languages for Non-Expert Robot Programmers. IEEE Access 7:17535--17550 (2019)
14. D. Weragama & J. Reye. Analysing student programs in the PHP intelligent tutoring system. Int. Journal of Artificial Intelligence in Education, Vol. 24, no. 2, 162--188 (2014)