ne-Course for Learning Programming

José Figueiredo
Research Unit for Inland Development
Polytechnic of Guarda, Portugal
jfig@ipg.pt

Natália Gomes
Research Unit for Inland Development
Polytechnic of Guarda, Portugal
ngomes@ipg.pt

Francisco José García-Peñalvo
Computer Science Department
Research Institute for Educational Sciences GRIAL research group
University of Salamanca
fgarcia@usal.es

ABSTRACT

Difficulties in learning programming are a constant concern in engineering courses. In many research studies involving the learning programming must of the solutions presented, from the beginning of the first programming languages, was to apply different type of problems analysis. Literature relating to the understanding of nature of learning programming skills has been focused explicitly on the teaching methodology and few of them focus on abilities, characteristics and knowledge acquired over the life cycle of learning programming in each student. Most of the students enrolled in engineering courses, where programming is a crucial competence, never had the opportunity to develop skills of computational thinking. In this paper, we focus our work on the learning programming developing and applying a set of exercises where students with more difficulties can express and develop their skills in computational thinking. In order to understand some programming students difficulties we have create a set of exercises, and apply it to a pre-programming course, that allows teachers to understand how students analyse and comprehend aspects such as visualization, spatial interpretation and physical manipulation. This paper also reports on results obtained from a class experiment where Memory Transfer Language was used by students to learn programming. All the exercises must be resolved without any type of technology, designed as a ne-course (no electronic course) for learning programming.

CCS Concepts

• Social and professional topics ~ Computing education • Social and professional topics ~ Computing education programs.

Keywords

programming education, introductory programming, cs0, cs1, learning programming, teaching programming.

1. CONTEXT AND MOTIVATION

Learning to program, generally considered hard, is a concern in all course of engineer. This phenomenon is universal and learning problems are not course, school or country specific.

TEEM'16, November 02 - 04, 2016, Salamanca, Spain

ACM 978-1-4503-4747-1/16/11…$15.00
DOI: http://dx.doi.org/10.1145/3012430.3012572

Since the appearance of the first programming languages this problem is been studied. There are numerous studies with the main reflection of the difficulties of solving programming problems. In this sense, the analysis of several studies, such as those conducted by the Natural Programming Project and Psychology of Programming Interest Group, among others, in some way, can contribute to demystify this problem. This studies can help teachers to understand what are the students difficulties, the reasons or the best tools, methods or technologies to improve learning programming [1].

The need for research in education, according to [2], appears when we want to better understand the operation of a particular educational situation, and we intend to answer the many questions we put on how to improve the way we act. Specifically, and according to [2], research should be done in education to perform the following actions:

- Responding to the need to meet and improve a particular educational reality.
- Using new methods in teaching and analyse the effectiveness of the application of these methods in order to improve an educational reality.
- Assess the situation studied and analyse the causes that led to a particular diagnosis.
- Generalizing conclusions that may affect others.

Our main motivation for the development of this work is to understand what difficulties students have, which factors most influence their learning programming process, which tools and/or methods or technologies can be used to reduce problems in the teaching / learning of the initial programming course. Develop a new learning environment of programming to help students to overcome their difficulties.

This paper is an attempt to demonstrate the importance to recognize in the first beginning of learning programming the difficulties that student may have and analyse the effectiveness of the methods we propose to implement. Our aim is to provide a new contribution in the learning programming area helping to discuss and demystify the principles of learning programming and to carry out and analyse the preliminary experiments with a set of exercises/methods.

2. BACKGROUND & RELATED WORK

The programming teaching is quite recent compared to other areas, such as mathematics and physics. The programming teaching requires a different methodology of other subjects. Theories as active learning, learning by doing, peer assisted learning, or peer instruction, with good results in several areas of education, also have confirmed results in this area [3]–[6]. Many other methodologies and techniques may be combined and should be applied and articulated in different environments and situations according to the needs and interests of each student. We must remember that students chooses to solve a problem will be different

from another student, and what one student takes away from an experience may will be different from the others.

2.1 Best practices

A systematic review of approaches for teaching introductory programming and their influence on success [7], have demonstrate a positive effect on approval rates in introductory programming courses. In 2014 some authors have presented a systematic review of articles describing approaches for teaching introductory programming and their influence on success [7]. The authors of this paper have revealed relevant literature through the analysis of several research publications available in ACM and IEEE database from 1980 to 2014. The results of this important research have been accepted in numerous refereed journals: Transactions on Computing Education, SIGCSE, ITiCSE, ICER, SIGITE and ICALT. According to the authors, this paper describes the best practices for improving positive effect on approval success rates in introductory programming courses compared to traditional education programming. Some of the best practices to engage and improving students in learning programing are:

- collaboration: activities that encourage student collaboration either in classrooms or labs.
- contextualization: activities where course content and activities were aligned towards a specific context such as games or media.
- CS0: the creation of a preliminary course that was to be taken before the introductory programming course.
- grading schema: a change in the grading schema; the most common change was to increase the amount of points rewarded from programming activities, while reducing the weight of the course exam.
- group work: activities with increased group work commitment such as team-based learning and cooperative learning.
- peer support: support by peers in form of pairs, groups, peer mentors or tutors.
- support: an umbrella term for all support activities, e.g. increased teacher hours, additional support channels, etc.

Many other studies have been reported regarding the importance of Computational Thinking (see [8], [9] and [10]) for improving learning programming. The studies revealed that Computational Thinking is a fundamental skill not just for computer students but for everyone and should be used in all areas, a commonplace. In other perspective some authors defends the use of game-based learning [11] to improve students' cognitive abilities and expectations about learning programming.

Apart from the use of different practices it is important for teachers to understand the differences and know in their students' learning styles. This knowledge is an important aspect to consider in order to define and implement the best methodologies and practice strategies into teaching and learning programming activities [12]–[15].

3. Our proposal

3.1 Study group

In this research, a study was conducted to investigate and explore the views of students and the difficulties they faced in learning programming courses. The study involved a group of students of Computer Engineering from the Polytechnic of Guarda, Portugal. The Polytechnic of Guarda (IPG) is an institution of higher education located in the interior of the country.

Our study group has very special characteristics which may affect, in our opinion, the learning programming process:

- The course of computer engineering, IPG, is usually not the first choice of students, which in some circumstances affect students' motivation and engagement.
- Average grade, in recent years, is between 10 and 12 values.
- Students reveal some general difficulties in the area of CS.
- From our years of experiences, we have found that most of the students have very particular difficulties in terms of computational thinking.

3.2 Pre-Programming course (CS0)

According to characteristics of the IPG computer engineering students, we have created a free course of pre-programming for improving positive effect on approval success rates in learning programming. This course is designed to provide students with a set of computational thinking exercises to substantially improve their cognitive abilities. The course is not mandatory and will function with teacher recommendation. The course session planning activity is:

1. Follow and Give instruction.
2. Map Design.
3. Paper Folding and Origami.
4. Memory Transfer Language.
5. Parson Problems.

3.2.1 Follow and Give instruction

The use of this kind of exercises has as purpose to increase the development of students' cognitive reasoning abilities and spatial visualization, strongly associated with the characteristics necessary for programming [1], [16], [17].

Based on this methodology, which are also used to evaluate the ability of students to programming, we have developed exercises to work with students. Some examples:

Example number 1: Students should design on a paper what a student or a teacher describes.

- On a sheet of paper draw a square measuring approximately 5 cm. on its sides.
- Draw a small dot in the center of the square.
- Draw a line that starts at the top right corner to the bottom left corner, passed by the point.
- Draw a line that starts in the upper left corner to the bottom right corner, passing the point.
- Write your first name in the triangle below the center point.

Example number 2: It is also possible to practice from an image, see Figure 1, asking students to describe it through the design of others images.

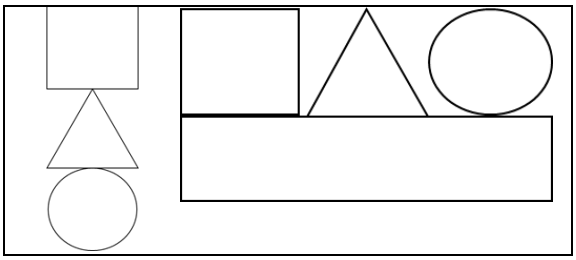

Figure 1 - Examples for follow and give instruction.

3.2.2 Map Design

With the use of this type of exercises we aim to develop students' capacities in planning, designing and describe in terms of specific characteristics in a concrete situation. Studies have demonstrated the relationship between the style and the level of detail in the description and construction of a map with the objectives of a programming course [1]. These activities include exercises for the student to move from point A to point B, within our school for example. This type of activity also includes the design and / or the representation of a path in a map. In this exercise we will evaluate the level of detail and clarity in the resolution.

3.2.3 Paper Folding and Origami

Origami and / or paper folding [16]–[18], [19], [20] is a Japanese secular art widespread throughout the world, known for the development of features, such as: visual and spatial perception, fine motor coordination, memory, relieving stress and tension, patience and persistence; self-confidence, logical thinking and attention and concentration. There are thousands of examples from the simplest to the most complex, of various categories, which can be used according to characteristics and likings of each.

Paper folding, in particular the Punched Holes, is frequently used to investigate the spatial visualization skills. In our case we want to use this activity for the development of student's capacity by solving various exercises. In this type of exercise students should imagine that is folding and unfolding paper. In each of the left and right drawing figures there are problems, see Figure 2. The figures at the left represent a square piece of paper being folded, and the last of these figures has one or two small circles drawn on it to show where the paper has been punched. The right figure shows the location of the holes when the paper is unfolded.

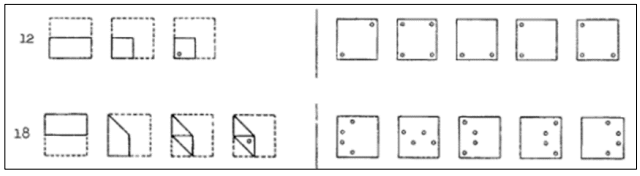

Figure 2 - Examples Punched Holes, adapted from [20]**.**

3.2.4 Memory Transfer Language

The used of Memory Transfer Language (MTL) exercises; allow us to overcome some problems detected in the construction of knowledge in early learning programming, particularly in the representation of variables and assignment statements. The methodology used to implement this kind of exercises was based on the representation of instructions in the computer memory. The construction of MTL exercises have been designed according to our experience in the teaching programming.

For the development of this set of exercises we also have analyse the work of Leonard Mselle and Hashim Twaakyondo, [21], where again, it is said that programming is a difficult concept to teach and learn. Related research has showed that concepts can be confused and abstract for all novices programmer's and the most difficult topic for students understanding is the abstract concepts involving the role of variable position in computer memory.

To determine the impact of MTL in aiding novice programmers to pursue their programming lessons without and with the intervention of a teacher, a class experiment, was conducted where examination results from two-phase experiment were statistically compared. To realize the experiment three exercises, according from the work of [22], have been design and applied, see Figure 3, 4 and 5.

Figures 3, 4 and 5, are examples of these exercises.

```
Program 1
begin
        int x, y
        read x
        read x, y
        write y, x, y
end

What is the expected output
 if you enter the values 3, 6, 9?
```

Figure 3 - Example 1 for MTL.

```
Program 2
begin
        int x, y
        read x
        read x
        read y
        write y, x, x
end

What is the expected output
 if you enter the values 3, 6, 9?
```

Figure 4 - Example 2 for MTL.

```
Programa 3
begin
        int x, y
        x = 5
        y = x
        x = x + 5
        y = x + 5
        write x, y
end

What is the expected output?
```

Figure 5 - Example 3 for MTL.

Once the student has completed the exercise, he or she must write, in a diagram previously define (Figure 6), the instructions executions.

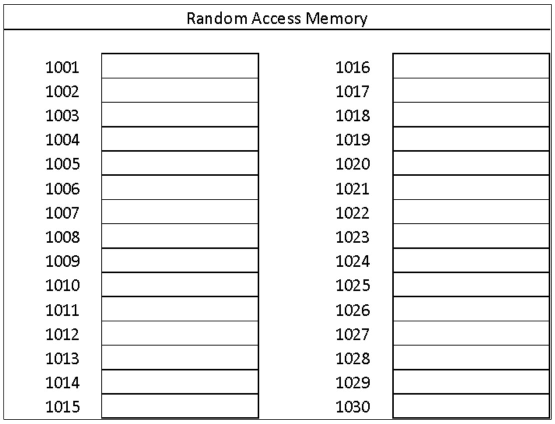

Figure 6 - Output representation for MTL exercises.

3.2.4.1 MTL class experiment

The experiment was carried out to test the comprehension that students may have between variable and computer memory thought the used of MTL exercises. As already referred the class experiment employs three exercises, questions exposed in Figure 3, 4 and 5.

To test the hypothesis that MTL can facilitate students pursue their learning programming classes a sample of 35 first year students of the IPG was used in the experiment. Students learning 'introduction to programming' for the first time in the computer engineering, academic year 2014/015, constituted the sample for this experiment.

Before the beginning of the experiment, a two-phase experiment, the teacher held a class with all students where they were briefed about computer programs and programming concepts such as variable, basic data types and computer memory representation.

3.2.4.2 First examination

Answer to the three exercises, of 35 students, to the first phase of the experimental examination, are summarized in Table 1. Correct answers have been assigned as 1 and incorrect answers as 0. As demonstrated in Table 1, only 20.0% of the students have correct answers and just only one student hit the three exercises.

Table 1 - Students answers - first analysis.

Programa 1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Programa 2	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Programa 3	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	1	0	0

3.2.4.3 Second examination

On a second phase of the experiment and after showing the results to students it was resolved and explained a set of similar exercises to clear their doubts. After clarification of the doubts it was proposed to the students to repeat the exercises. Table 2 shows the results on this second phase of the experiment.

Table 2 - Students answers - second analysis.

Programa 1	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1
Programa 2	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1
Programa 3	1	0	1	1	0	1	1	1	0	1	0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	1

As can easily be seen, in Table 2, the results were significantly better. The results increased from 20.0% to 81.0% of correct answers. Using final scores, between the first phase of the experiment and the second, where the experimental was support by the teacher, to learn programming results suggest a significant difference statically. The totality of correct answers increases from 1 student to 23 students. The capacity of student's mental abstraction, after the use of this activity, was improved.

3.2.5 Parson Problems

The last activity course is based on Parson Problems. According to [23]–[25] one way to learn and practice introduction to programming is using Parson Problems. Parson's problems are assignments for learning programming where the student has to select, order, and indent code fragments. The goal could be as example to construct a program which fulfils the task of an assignment. These assignments are great for an initial phase of the learning programming because students do not make syntax errors. In Figure 7, we can see an example of Parson Problem.

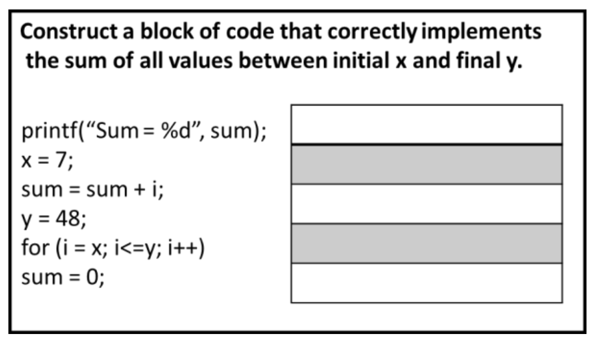

Figure 7 - Parson Problem example.

4. Discussion and Conclusion

With this work we intend to present our idea in open the discussion on learning programming. The objective of this study was to test the impact of different activities (Follow and Give instruction; Map Design; Paper Folding and Origami; Memory Transfer Language and Parson Problems) when used as a learning programming tool without the intervention of any electronic component (technology).

Specifically, MTL has been proved and initial results are encouraging though far from conclusive. There are, obviously, some shortcomings in this study. The sample size is too small to justify generalization.

The purpose of this ne-course is intentional, since it was our goal that students handle and solve the exercises manually, like board games, where students explore with pleasure, without fear of making mistakes and where teacher-student relationship and confidence can be improved and enhanced. As computer science teacher we must mention that the use of technology is also very

important to understand and analyse some of the activities done by students especially with a bigger sample.

Future work will focus on the development of an Intelligent Tutoring Systems to help teachers to manage exercises and students' scores in initial programming learning.

Nowadays where technology dominates all fields of our activity and interpersonal relationships are forgotten, we believe that it's still important to see the face and expressions of students in solve programming problems. We want to feel the atmosphere and excitement in problem solving.

Finally, we would like to highlight the speech Rita Pierson, in "Every kid needs a champion", TED Talks Education, May 2013:

> "You know, kids don't learn from people they don't like."

5. REFERENCES

[1] S. Fincher, B. Baker, I. Box, Q. Cutts, M. De Raadt, P. Haden, J. Hamer, R. Lister, M. Petre, A. Robins, K. Sutton, D. Tolhurst, and J. Tutty, "Computer Science at Kent programming courses," no. 1, 2005.

[2] R.-A. M. González, *La investigación en la práctica educativa: Guía metodológica de investigación para el diagnóstico y evaluación en los centros docentes.* Ministerio de Educación Cultura y Deporte, Centro de Investigación y Documentación Educativa, 2007.

[3] E. Nuutila, S. Törmä, P. Kinnunen, and L. Malmi, "Learning Programming with the PBL Method - Experiences on PBL Cases and Tutoring."

[4] J. O'Kelly, J. P. Gibson, J. O'Kelly, and J. P. Gibson, "RoboCode & problem-based learning," in *Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education - ITICSE '06*, 2006, vol. 38, no. 3, p. 217.

[5] L. Porter, D. Zingaro, and R. Lister, "Predicting student success using fine grain clicker data," in *Proceedings of the tenth annual conference on International computing education research - ICER '14*, 2014, pp. 51–58.

[6] R. San-Segundo, J. M. Montero, J. Macías-Guarasa, R. Córdoba, and J. Ferreiros, "Indianapolis, IN 35 th ASEE/IEEE Frontiers in Education Conference S2D-17 Automatic Tools for Diagnosis and Feedback in a Project Based Learning Course."

[7] A. Vihavainen, J. Airaksinen, and C. Watson, "A systematic review of approaches for teaching introductory programming and their influence on success," *Proc. tenth Annu. Conf. Int. Comput. Educ. Res. - ICER '14*, pp. 19–26, 2014.

[8] J. M. Wing, "Computational Thinking: What and Why?," 2010.

[9] J. M. Wing, "Computational Thinking," 2012.

[10] F. J. Garcia-Peñalvo, "What Computational Thinking Is," *J. Inf. Technol. Res.*, vol. 9(3), v–vi, no. October, 2016.

[11] J. Trybus, "Game-Based Learning: What it is, Why it Works, and Where it's Going," *New Media Institute White Papers*. [Online]. Available: http://www.newmedia.org/game-based-learning--what-it-is-why-it-works-and-where-its-going.html.

[12] A. Alharbi, D. Paul, F. Henskens, and M. Hannaford, "An Investigation into the Learning Styles and Self- Regulated Learning Strategies for Computer Science Students."

[13] L. Cândida, S. Carmo, M. J. Marcelino, and A. J. Mendes, "The Impact of Learning Styles in Introductory Programming Learning."

[14] L. Carmo, F. Pereira, A. Gomes, and A. Mendes, "Learning styles and problem solving strategies."

[15] A. Jimoyiannis, "Using SOLO taxonomy to explore students' mental models of the programming variable and the assignment statement," no. 4, pp. 53–74, 2011.

[16] N. E. Study, "An Overview of Tests of Cognitive Spatial Ability," *66th EDGD Mid-Year Conf. Proc.*, p. 6, 2012.

[17] Simon, S. Fincher, A. Robins, B. Baker, I. Box, Q. Cutts, M. De Raadt, P. Haden, J. Hamer, M. Hamilton, R. Lister, M. Petre, K. Sutton, D. Tolhurst, and J. Tutty, "Predictors of success in a first programming course," *Proc. 8th Austalian Conf. Comput. Educ. - Vol. 52*, pp. 189–196, 2006.

[18] S. Cooper, K. Wang, M. Israni, and S. Sorby, "Spatial Skills Training in Introductory Computing," *Proc. Elev. Annu. Int. Conf. Int. Comput. Educ. Res.*, pp. 13–20, 2015.

[19] Z. Falomir, "Towards A Qualitative Descriptor for Paper Folding Reasonin."

[20] A. J. Jaeger, J. Wiley, J. Pellegrino, K. Zinsser, M. Stieff, and T. Moher, "What Does the Punched Holes Task Measure?"

[21] T. Mselle LM and H., "The impact of Memory Transfer Language (MTL) on reducing misconceptions in teaching programming to novices," *Int J Mach. Learn Appl*, vol. 1, no. Art. #3, p. 6, 2012.

[22] A. Jimoyiannis, "Using SOLO taxonomy to explore students' mental models of the programming variable and the assignment statement," *Themes Sci. Technol. Educ.*, vol. 4, no. 2, pp. 53–74, 2011.

[23] B. B. Morrison, L. E. Margulieux, B. Ericson, and M. Guzdial, "Subgoals Help Students Solve Parsons Problems," *Proc. 47th ACM Tech. Symp. Comput. Sci. Educ.*, pp. 42–47, 2016.

[24] B. J. Ericson, "Adaptive Parsons Problems with Discourse Rules," *Icer '14*, pp. 145–146, 2014.

[25] P. Denny, A. Luxton-Reilly, and B. Simon, "Evaluating a new exam question: Parsons problems," *Proc. fourth Int. Work. Comput. Educ. Res.*, pp. 113–124, 2008.