An Overview of Computer Programming Teaching Methods

Sanja Mohorovičić
Faculty of Maritime Studies
University of Rijeka
Studentska ulica 2, 51000 Rijeka, Croatia
sanja@pfri.hr

Vedran Strčić
Department of Informatics
University of Rijeka
Omladinska 14, 51000 Rijeka, Croatia
vstrcic@inf.uniri.hr

Abstract. *This paper researches different computer programming teaching methods in courses in higher education. Computer programming is among the most challenging subjects in computer science curriculum, and the one that many students find difficult to grasp, hence it is very important to select an appropriate teaching strategy that will provide students with the most efficient learning environment. The paper gives an overview and comparison of some of the existing teaching methods and addresses the benefits and problems associated with their application.*

Keywords. computer programming, teaching strategy, teaching methods

1 Introduction

All computer based technology that we use today is controlled by computer programs that are written to define its behaviour. Whether it is our personal computer, a tv, or an airplane, they all have a single or a set of computer programs integrated within them. Computer programs define and describe available functions of devices they are integrated in and provide us with a way to control them.

One of the definitions of a computer program is given through a structured programming principle that describes it as a set of tasks. "Any task that is too complex to be described simply is broken down into a set of smaller component tasks until the tasks are sufficiently small and self-contained enough that each is easily understood." [1] Defined in boundaries of a personal computer a program is "a sequence of instructions written to perform a specified task for a computer" [2]. "Computer programming is the process of designing, writing, testing, debugging / troubleshooting, and maintaining the source code of computer programs." [3]

Computer programming courses are a part of many universities' curriculums, and among the most important subjects for a computer science students. Many authors [4] [5] [6] [7] [8] agree that learning to program presents a challenge for many students, and that many of them find programming concepts difficult to grasp. The evidence are dropout and failure rates in introductory programming courses at the university level [5].

Considering the challenge that computer programming represents to the students, we can assume that finding and implementing an appropriate teaching strategy will be one of the crucial factors in students' success in mastering the course content.

This paper provides an overview of some of the existing programming teaching methods, and compares the results obtained by their application.

2 Specific programming issues

Programmers are human beings, influenced by factors like confidence, comfort and satisfaction. As such they require more than just familiarity with a language semantics to be effective [11]. Learning programming will accordingly require more than just acquiring skills, it will be greatly affected by students self-efficacy that will influence the use

of cognitive strategies while solving problems, the amount of effort expended, the type of coping strategies adopted, the level of persistence in the face of failure, and the ultimate performance outcomes [5].

Computer programming requires the use of complex cognitive skills, such as reasoning, problem solving and planning [6]. A programmer forms abstract representations of a process, expresses them in the form of logic structures, and finally translates them into correct code using the formal language [5].

The three primary pedagogical goals in teaching a programming language are therefore covering the language's syntax, developing program design skills and creative thinking. The selection of the most suitable programming language and the teaching approach are presented as two fundamental issues related to teaching programming [10].

3 Teaching strategies

3.1 Learning styles and motivation

Students' learning styles and motivation are factors that affect the success of any learning process they engage in. Learning programming is, as such, also affected by those factors.

"Learning style is a preferred learning mode in which students respond to and use stimuli in the context of learning." [6] Some students are very visual, some are more auditory or kinaesthetic [9]. Selecting a particular learning style, or a particular form of motivation, may allow a student to acquire programming skill quickly and easily. Opposed to that, if a student adopts the wrong style or lacks the motivation, he may find learning to program difficult [4].

According to one classification of learning styles [4], learning is divided into deep and surface approach. Deep learning refers to gaining understanding of a topic, while surface learning concentrates on memorising the facts. In programming, surface learning can be used for memorising the language's syntax, but deep learning is crucial, in addition to surface learning, for gaining a true understanding of programming logic and consequently a true competence in programming.

Motivation, as another important factor that affects learning efficiency, can be extrinsic, intrinsic or social [4] [12]. Extrinsic motivation comes from expected external rewards, such as financial benefits, intrinsic motivation comes from within, and social from a desire to please a third party (family, teacher, friends, etc.).

The two factors that affect the extent of motivation are expectancy and value [12]. A value represents the reason why student values success, e.g. future employment, and expectancy is a student's own expectation of success. If student doesn't value the success, or doesn't expect to pass the programming course, he might not be motivated to engage in course assignments.

The teacher's task is not only a transmission of knowledge, he also bears the role of a motivator, making sure the students really do engage in assignments he has devised.

3.2 Approach to teaching programming

Teaching (instructional) method is "an approach used by educators in course content deliveries. It is an educational approach for revolving knowledge into learning which focusing on the "how to" in delivery of training" [6].

Information and communication technologies (ICT) teaching, and consequently the teaching of computer programming, is relatively new problem domain, compared to teaching of already established subjects, such as mathematics or physics. As such its teaching methodology isn't as firmly formulated, which results in the fact that most teachers use their own blend of methods, where one of those methods is represented dominantly [13].

One of the classifications [13] divides computer programming teaching methods according to their orientation: statement-oriented, using as a tool, software technology-oriented, task type-oriented, language-oriented, action-oriented and sample task-based. By this classification different methods cover the chosen programming language to the different extents, and teach its elements in a different order depending on the set goals. Statement oriented methods, as such, see the programming language as set of statements, and all individual elements are then taught in a certain fixed order. Task type oriented methods introduce practical problems and then present language elements

in such order in which they are needed to solve those tasks. "Using as tool" methods introduce a programming language only to a necessary extent, considering a different primary goal, such as database teaching.

Teaching programming can also be divided on bottom-up and top-down approach [14]. Bottom-up approach primarily focuses on teaching the details of syntax and individual programming language elements first. After individual elements have been taught, more complex constructs are considered. Top-down approach starts with understanding the abstractions regardless of their physical implementation. After students understand these abstractions and their purpose, implementations are being taught.

4 Programming teaching methods

This section presents an overview of some of the existing programming teaching methods. Each method is described, and some results obtained by its application are presented.

4.1 Problem-based learning

Problem-based learning (PBL) method [15] [16] focuses on students' own engagement in problem solving. This approach is centered around the types of problems that professionals in the field encounter on daily basis. It develops higher order thinking, disciplinary knowledge and practical skills, by facing students with a problem situation, and assigning them an active role of problem solvers.

PBL is implemented by seven steps method [15] that encourages learning by encompassing students' prior knowledge about the presented topics, practical problem situations, and the required students' subsequential elaboration on materials they have learned.

Students mainly work in groups. They indentify and discuss the problem, then make a list of what they already know, and what they still need to learn. That is followed by establishing the learning goals, after which the students independently study all the required material. The students then reconvene to discuss the case, and attempt to apply what they learned in order to solve the problem. Finally, they summarise their work and elaborate on their solutions.

Different models of PBL also exist [16]. They vary from the simple problem-based approach where lectures are presented normally, but problems are introduced to motivate the students and demonstrate a theory, to full PBL models where the problems guide entire learning process. Students can also work individually or in groups.

Few researchers [15] concluded that while there is no evidence of improvement in immediate students' examination results, PBL encouraged increased retention of knowledge over the period up to several years, and students' had better results in follow up courses, compared to their colleagues who attended traditional introductory programming courses.

It was also concluded [15] that PBL enhances students' communication skills, creative thinking, motivation and responsibility.

4.2 Puzzle-based learning

Puzzle-based learning (PZBL) [17] [18] [19] aims to teach students critical thinking and problem solving techniques.

Puzzles should generally be easy to state and remember. A problem should present a challenge to problem solver, having no obvious solution, but still showing the promise of resolution [19].

In computer programming courses, the process of puzzle solving is conducted in a few steps [18]. Firstly, the problem is presented to the student according to content that is being taught. A complete program solution is then divided into number of program puzzle pieces depending on the desired difficulty. Minimum size for a puzzle piece is one complete line of code. Student then attempts to reconstruct the program, by selecting the correct program pieces in the correct order, after which his success is evaluated. This procedure can be guided by a teacher or by an automated system [18].

Results of an empirical study [17], conducted in one introductory programming course where PZBL method was implemented, showed that students interest and scope for active participation in the programming course was significantly increased.

4.3 Pair programming

Pair programming (PP) [20] [21] is programming style in which two programmers work side by side, on the same code, at the same computer. They both continuously participate in the design, development and testing of that code.

The programmers that work as a pair assume two roles, one of the driver and second of the navigator. The driver is in control, typing the actual code, while navigator observes his work, watching for potential errors, offering suggestions, and devising alternatives. The roles are, at the regular intervals, switched, ensuring that both programmers continually provide the same amount of effort.

Researchers [20] suggest that the software, that was developed using pair programming method, is usually completed in less time then when individual students code it, is better designed and has less errors. Students are also better motivated to stay on task, have more confidence in their solutions, and show a positive attitude towards collaboration.

Some studies [21] have shown that, despite all its benefits, pair programming can sometimes be irritating and exhausting. It is suggested that a difference in skill level between paired students strongly affects their collaboration, and that an appropriate care when organizing the pairs of students is necessary.

Virtual pair programming is a form of pair programming that eliminates the need of two programmers sitting at the same physical location. Their collaboration is realized by using online tools that integrate desktop sharing and real time communication. A study [22] showed that it is an effective pedagogical tool for flexible collaboration.

4.4 Prerecorded lectures

Prerecorded lectures (PL) [23] in form of carefully prepared and technically focused multimedia recordings that are kept online, comprise of narrated slides from the lecture notes, and are meant to supplement the conventional lectures.

PL aim to confront the issue of limited semester timetable, helping those students who fail to absorb the presented course material at sufficient rate. They provide students with a possibility to review all the concepts they may have missed or not understood well enough while attending the original lectures. Unlike classic textbooks, PL better separate the important concepts from less important ones, and can be more easily refined over time to better stress those particular concepts that students had issues with over past semesters [23].

In a research [23], that was conducted among students that used 5 to 15 minutes long PL, each focusing on particular topic of the course, it was found that PL made no significant difference on students final grades. It was also found that part of the students, who have seen the PL only as an optional material, have never accessed any of the PL online. Overall, however, students' feedback on the PL was very positive, and most of them stated that PL helped them better understand some programming concepts.

4.5 Game-themed programming

Game-themed programming (GTP) [24] is a teaching approach which integrates simple interactive graphic programs into introductory programming courses. The aim of this approach is to make students learn about abstract programming concepts by exploring and programming small game applications.

Primary goal of GTP is not teaching students to build computer games, but teaching them programming concepts through understanding how the games work. Any assignment that would, in a classical approach, be a simple console assignment, is reworked so it becomes a simple game assignment, and that the devised game logic is based on targeted programming concept [24].

Game-themed assignments, that are given to students, include the descriptions of the tasks that need to be completed and starter projects that consist of provided graphics and functional user interaction modules. Students are then required to complete the assigned projects by filling in the missing relevant concepts [24].

A study [24] revealed that success rates in GTP classes were higher than in normal classes. Students reported that they had to spent more time to understand the game assignments than they would require to understand classical console assignments, but when they finally understood them, their motivation and enthusiasm increased.

5 Comparison of methods

All of the presented programming teaching methods aim to enhance the quality of existing programming courses with a goal of making the programming concepts easier to grasp.

The main commonality that all these approaches share is an attempt to increase students' motivation in order to reduce their frustration when faced with challenging concepts and abstractions, and to encouraging them to increase their engagement in tasks that teacher has devised for them.

The methods differ in their focus. While some focus on how the programming concepts are presented to the students and how the course materials are delivered (e.g. prerecorded lectures), other try to find alternative ways in which students can work with those concepts (e.g. game-themed programming).

Teachers task is to select an appropriate teaching method, or a blend of methods, to devise assignments for students accordingly, and after presenting the concepts in the way he/she has chosen, to shift the focus on students, motivating them to engage in assignments he has devised.

6 Conclusion

This paper overviews some of the existing programming teaching methods, addresses the results obtained by their application, and compares some of the common features they share.

The challenge that teaching and learning computer programming presents, has encouraged the design and implementation of various new and innovative computer programming teaching methods. The presented methods aim to improve the students' success rates by increasing their motivation and encouraging the greater self-engagement, not only in assignments provided within a course, but also in further exploration of the programming challenges outside the assignments' boundaries.

References

[1] Liberty J, Rao S, Jones B: Sams teach yourself C++ in one hour a day, Sams Publishing, USA, 2009.

[2] Stair R, Reynolds G: Principles of Information Systems, Course Technology, USA, 2003.

[3] Meijers A W M, Gabbay D M, Thagard P, Woods J: Philosophy of Technology and Engineering Sciences, North Holland, Amsterdam, 2009.

[4] Jenkins T: On the Difficulty of Learning to Program, Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, Lough- borough, UK, 2002, pp. 53-58.

[5] Wiedenbeck S, LaBelle D, Kain V N R: Factors Affecting Course Outcomes in Introductory Programming, Proceedings of the 16th Workshop on Psychology of Programming, 2004, pp. 97-109.

[6] Tie H H, Umar I N: The Impact of Learning Styles and Instructional Methods on Students' Recall and Retention in Programming Education, Proceedings of the 18th International Conference on Computers in Education, Putrajaya, Malaysia, 2010, pp. 191-195.

[7] Radošević D, Lovrenčić A, Orehovački T: New Approaches and Tools in Teaching Programming, Proceedings of the 20th Central European Conference on Information and Intelligent Systems, Varaždin, Croatia, 2009, pp. 49-57.

[8] Mohorovičić S, Tijan E: New Technologies in Teaching University Level Programming, Proceedings of the 33rd International Convention on information and communication tecnology, electronic and microelectronics (MIPRO 2010), Vol. IV. CE, Croatia, 2010, pp. 268-272.

[9] Hu M: A Case Study in Teaching Adult Students Computer Programming, Proceedings of the 16th Annual NACCQ, Palmerston, North-New Zealand, 2003, pp. 287-290.

[10] Al-Imamy S, Alizadeh J: On the Development of a Programming Teaching Tool: The Effect of Teaching by Templates on the Learning Process, Journal of Information Technology Education, Vol. 5, 2006, pp. 271-283.

[11] Smith R B, Ungar D: Programming as an Experience: The Inspiration for Self, Object-Oriented Programming 9th European Conference, Vol. 952, Denmark, 1995, pp. 303-330.

[12] Jenkins T: Teaching Programming - A Journey from Teacher to Motivator, Proceedings of 2nd Annual LTSN-ICS Conference, 2001, pp. 65-71.

[13] Papp-Varga Z, Szlavi P, Zsako L: ICT teaching methods - Programming languages, Annales Mathematicae et Informaticae, Vol. 35, 2008, pp. 163-172.

[14] Reek M M: A top-down approach to teaching programming, Proceedings of the twenty-sixth SIGCSE technical symposium on Computer science education, New York, USA, 1995, pp. 6-9.

[15] Nuutila E, Torma S, Malmi L: PBL and Computer Programming - The Seven Steps Method with Adaptations, Computer Science Education, Vol. 15, Issue 2, 2005, pp. 123-142.

[16] Wu Y: Applying a hybrid problem-based learning method to the teaching of computer programming, The China Papers, Issue 6, 2006, pp. 63-66.

[17] Merrick K E: An Empirical Evaluation of Puzzle-Based Learning as an Interest Approach for Teaching Introductory Computer Science, IEEE Transactions on Education, Vol. 53, No. 4, 2010, pp. 677-680.

[18] Yoneyama Y, Matsushita K, Mackin K J, Ohshiro M, Yamasaki K, Nunohiro E: Puzzle Based Programming Learning Support System with Learning History Management, Proceedings of the 16th International Conference on Computers in Education, 2008, pp. 623-627.

[19] Falkner N, Sooriamurthi R, Michalewicz Z: Puzzle-Based Learning for Engineering and Computer Science, Journal Computer, Vol. 43, Issue 4, USA, 2010, pp. 20-28.

[20] Zacharis N Z: Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course, IEEE Transactions on Education, Vol. 54, No. 1, 2011, pp. 168-170.

[21] Chaparro E A, Yuksel A, Romero P, Bryant S: Factors Affecting the Perceived Effectiveness of Pair Programming in Higher Education, Proceedings 17th Workshop of the Psychology of Programming Interest Group, 2005, pp. 5-18.

[22] Zacharis N: Evaluating the Effects of Virtual Pair Programming on Students' Achievement and Satisfaction, International Journal of Emerging Technologies in Learning (iJET), Vol. 4, No. 3, 2009, pp. 34-39.

[23] Smith G, Fidge C: On the efficacy of prerecorded lectures for teaching introductory programming, Proceedings of the tenth conference on Australasian computing education, Vol. 78, 2008, pp. 129-136.

[24] Sung K, Hillyard C, Angotti R L, Panitz M W, Goldstein D S, Nordlinger J: Game-Themed Programming Assignment Modules: A Pathway for Gradual Integration of Gaming Context Into Existing Introductory Programming Courses, IEEE Transactions on Education, Vol PP, Issue 99, 2010, pp. 1-12.