Experience Teaching Introductory Programming Courses

Liguo Yu, Indiana University South Bend, South Bend, IN, USA

ABSTRACT

This paper describes the teaching methods and teaching experiences of an instructor who has been teaching computer science freshmen for more than ten years. Problem-based learning has been used in introductory programming courses to enhance the quality of learning. The common problem in introductory programming courses is that instructors quite often focus mainly on programming language syntax and usage but ignore the real world applications of these language features. Accordingly, students are likely to find out that it is difficult to apply their programming skills on solving complex problems in their high level courses. Through introducing problem-based learning in introductory programming courses, students not only could obtain experience of dealing with complex problems, but also could practice other soft skills, such as team working skills, which could better prepare them for junior and senior level coursework and capstone projects. In addition, other philosophy issues related to teaching introductory courses are discussed and tentative solutions are presented.

KEYWORDS

Computer Science Education, Introductory Programming Education, Lab Assignments, Problem-based Learning, Projects, Teaching Philosophy, Teaching Quality, Teamwork

1. INTRODUCTION

Programming is a fundamental skill for computer science students (Davies et al., 2011). Usually, programming is the first course offered to computer science freshmen. It might take one or two semesters for students to learn their first programming language, which could be either C++ or Java, as can be seen in most higher education institutions. Because in most cases, this will be the first time that students get to know a programming language, most instructors consider teaching how to use a language a higher priority in these courses. Accordingly, students could focus more on language syntax and usages and might not get more opportunities to practice their analytical skills and problem-solving skills.

The drawbacks of focusing mainly on language aspects in introductory programming courses are not commonly understood in computer science education community (Huet et al., 2004). Therefore, they are described here. First, students could understand one programming language well but might be hard to know how to use it properly. Second, given the diverse functions provided by a programming language, students might not know which language feature should be used to solve a problem. Third, students might not get a chance to appreciate the complexity of real-world problems. Studies show that there is a great need to change this traditional teaching approach (Ngo, 2015).

Within the computer science community, it is commonly agreed on that lecture-based learning is not the most effective approach in ensuring the teaching quality and helping students achieve their
learning objectives. Problem-based learning (Barrows and Tamblyn, 1980; Jones, 2006), on the other hand, has become a major teaching and learning method used in computer science education. However, it is not an easy task to develop an introductory programming course based on problem-based learning, as in most institutions, the learning objectives of introductory programming course is about language education. Some educators even consider that it is not important or necessary to focus on problem solving in introductory courses. The logic of these beliefs is that we do not teach our kids how to write an essay in their first grade, instead they should be taught how to read. The problem with this analogy is that our college students are not elementary school kids and they are adults; we do not need to spend about 12 years to teach an adult a programming language as we teach English to K-12 students. Accordingly, we should not follow exactly the same teaching method used in elementary or secondary education.

Problem-based learning is a student-centered learning approach. The experience of solving real-world problems can help students acquire knowledge and/or skills that could better prepare them for high level courses and their future career (Hmelo-Silver, 2004; Kay et al., 2000). To reduce the gap between industry expectations and what the academia can provide, problem-based learning has been introduced in computer science and software engineering education (Richardson and Delaney, 2009; Armarego, 2002; Richardson et al., 2011). Table 1 summarizes the differences between instructor-centered learning and problem-based learning in computer science and software engineering education (Yu and Adaikkalavan, 2016).

In this paper, we discuss the undergraduate introductory programming course offered at Computer Science Department, Indiana University South Bend. We discuss how we apply problem-based learning in the course. We analyze the challenges we have been facing and present our experiences to deal these challenges. Specifically, original assignment examples are provided and the readers are free to reference and use these materials.

Teaching introductory courses and interacting with freshmen or sophomores is always critical in higher education. Besides teaching methods and teaching techniques, there are other teaching philosophy issues, such as instructor-student relations, need to be addressed carefully. Our experience and suggestions related to freshman/sophomore education are also discussed.

The remaining of the paper is organized as the followings. Section 2 reviews current status of teaching introductory programming courses in higher education, including teaching approaches and teaching techniques. Section 3 describes our problem-based learning approach and teaching experience. Section 4 presents other philosophy issues in teaching introductory courses. Conclusions are in Section 5.

Table 1. Comparison of instructor-centered learning and problem-based learning (Yu and Adaikkalavan, 2016)

<table>
<thead>
<tr>
<th>Instructor-centered learning</th>
<th>Problem-based learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge is passed from instructor to students</td>
<td>Knowledge is learned through solving real-world problems</td>
</tr>
<tr>
<td>Skills are educated through lectures</td>
<td>Skills are learned through practicing</td>
</tr>
<tr>
<td>Computer science knowledge is presented</td>
<td>Computer science knowledge is applied</td>
</tr>
<tr>
<td>Unique solution is provided</td>
<td>Many solutions are discussed</td>
</tr>
<tr>
<td>Instructor is the customer who has the clear and fixed requirements</td>
<td>Interacting with real-world customers whose requirements could be vague and variable</td>
</tr>
<tr>
<td>Problems from the textbook may be outdated or unrealistic</td>
<td>Solving the real-world problems that students can relate to</td>
</tr>
<tr>
<td>Failure of the assignments indicates failure of the learning objective</td>
<td>Failure, which is part of the learning process, is allowed, and lessons could be learned</td>
</tr>
</tbody>
</table>
Developing Communities of Practice to Prepare Software Engineers with Effective Team Skills
www.igi-global.com/chapter/developing-communities-of-practice-to-prepare-software-engineers-with-effective-team-skills/102320?camid=4v1a

Using Blended Learning Approach to Deliver Courses in An Engineering Programme
www.igi-global.com/article/using-blended-learning-approach-to-deliver-courses-in-an-engineering-programme/163289?camid=4v1a