Experiences in Using an Automated System for Improving Students' Learning of Computer Programming*

M. Choy, U. Nazir, C.K. Poon, and Y.T. Yu

Department of Computer Science, City University of Hong Kong
{csmchoy, usmann, csckpoon, csytyu}@cityu.edu.hk

Abstract. Practical exercises and assignments are an integral part of programming course, which is a core basic skill required in computer science and best learned by doing. For effective learning, assignments have to be inspected and graded carefully but quickly so that students may benefit from the useful and timely feedback. This can be facilitated by automating the process using a specifically designed software system. PASS is such a system being developed in our department. A first version has been pilot run, with very encouraging responses from tutors and students. We observed that with this automated system, students can have their programs tested anytime, anywhere. They are encouraged to practice more without hesitation or embarrassment about their programming mistakes, and thus this learning environment can boost up their motivation to continue the practice of their programming skills. This paper aims at sharing the initial experiences we gained in using this system.

1 Introduction

Practical exercises and assignments are an integral part of courses in computer programming, which is a core skill required in Computer Science and best learned by doing. Calif and Goodwin [3] described their experiences of students who considered computer programming simply as a course of hurdle. There were severe motivation problems that inhibited them to practice more, and many students relied on memorization of program codes to pass the course.

From our experiences in teaching first year undergraduate students programming courses, we observed a significant discrepancy between the coursework marks and the programming skills actually gained by the majority of the class. It is becoming a common phenomenon for university students to submit a friend's work, a program with part of the code copied from others', a fusion of several programs, a joint effort group work, or even paid work as their own work in programming assignments. Using these tricks, getting a reasonable or even high mark might not be so difficult. Besides programming assignments, the other major contribution in assessment comes from the end-of-course examination. According to our statistics from the years 2000

[*] The work described in this paper was partially supported by Teaching Development Fund (project no. 6980041) from City University of Hong Kong.

R.W.H. Lau et al. (Eds.): ICWL 2005, LNCS 3583, pp. 267 – 272, 2005.

to 2003, the style of computer programming examination questions for first year undergraduate students is composed of writing small program segments (36.6%), tracing program for outputs (21.2%), completing a program statement or a program definition (20.4%), essay questions (10.6%), finding program errors (6.2%), and writing a complete program (5%). We note that it is rare to require students to write a complete program on their own. Thus, it is possible for students to get a good grade in the course even if they cannot write computer programs without assistance. These observations are similar to the findings in universities elsewhere [1, 2].

This year, we have introduced an automated programming assessment system, known as PASS, in several programming courses so that students may learn from the system by testing their programs. This way, we hope that students will feel that writing a correct program is within their reach and hence be encouraged to practice more on their programming skills instead of relying on plagiarism or code memorization to pass the course, thereby boosting their motivation in learning computer programming.

2 Using an Automated Programming Assessment System

A Web-based automated Programming Assignment aSsessment System (PASS) is currently used by several teaching staff at the Department of Computer Science, City University of Hong Kong. A preliminary prototype version [4] was initially built for demonstration purposes in early 2004, now it has been used in several programming courses. With PASS, everything is done on-line: setting up user accounts by the administrator, uploading exercises or assignments together with testing data by the tutor, downloading the exercises, and on-line program testing by the students.

For assignments, students are required to submit their programs to the system on line. Marking will start automatically when the tutor initiates the process by clicking a button. As soon as the assignments are marked, the tutor will receive the assessment report, and the students can get to know their results together with the feedback added by the tutor. Fig. 1 shows the interactions between a tutor and a student through the system. Detailed descriptions of this system can be found in [4]. PASS is used in laboratory sessions too. Each week, students will be writing 3-5 programs during the laboratory session. For laboratory exercises, there is no submission, but the system still provides test cases for selected exercises and a testing environment so that students can work with the exercises anytime, anywhere.

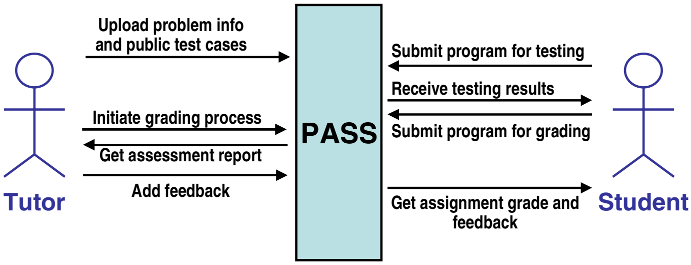
Upload problem info and public test cases
Initiate grading process
Get assessment report
Add feedback
Tutor
PASS
Submit program for testing
Receive testing results
Submit program for grading
Get assignment grade and feedback
Student

Fig. 1. Interactions between the tutor and student through PASS [4]

2.1 Using PASS for Assignment Assessment

When the problem given is an assignment, only a few test cases are made publicly available in PASS for students to test their programs before submission. These test cases belong to a subset of the complete set of test cases used in the assessment of this assignment. The rest of the test cases are hidden from the students in order to prevent any attempt to hard code the results into the submitted programs.

The automated testing environment makes possible the testing of students' programs with a lot more testing data than in the past when the programs were executed by hand. Hence, a more complete checking of correctness can be done. In addition, the assessment becomes more objective and consistent compared to manual marking.

2.2 Using PASS for Guidance to Learning

For laboratory exercises, the system will make available all test cases for students' own testing. Students can test their programs online, and use the instant feedback to improve their work and retest to check their progress. Students can continue to work on the exercises and have them checked by PASS even after their laboratory session.

We understand that there are many ways to define the logics to solve a programming problem. With the high degree of creativity of students, they might come up with a lot more ways to write a program than experienced programmers normally do. A large number of test cases are needed to test each program in order to uncover as many faults in their work as possible.

With the arrangement of weekly laboratory exercises and the automated testing of programs, PASS is able to help students in their learning of computer programming. Instead of waiting for the tutor to check their programs one by one, students can obtain instant feedback on the correctness of their programs. On a number of programming problems, by repeated usage of the system, students are able to develop better awareness of program correctness, and hence they will write programs more carefully. This has been observed by the tutors in our trial-run. By providing such continuing feedback to students who use PASS for their laboratory exercises, students are encouraged and motivated to practice more.

2.3 A Comparison Between Teaching Programming With and Without PASS

Before PASS was developed, during laboratory sessions, the tutor had to manually go through students' program codes to make sure they were correct. Testing programs with many test cases during a 1-2 hours laboratory sessions is not quite feasible. Therefore, usually the tutor would only use a few predefined test cases to test the students' programs. Sometimes the tutor might have to help students in debugging. With only a few test cases, it is likely that the program still contained errors even after being checked by the tutor. With this approach, not much could be done within the laboratory session, and it was almost always impossible to check everyone's work. Students would then be responsible to test their own programs with additional test cases they created, and consult the tutor after class if they encountered problems. Unfortunately, most students did not know how to create adequate test cases on their own. In this way, students had to be adequately self-motivated to continue working with the exercises outside the laboratory session. Without further supervision outside the class, it was not surprising to see that motivation usually dropped dramatically when students started to lag behind in the scheduled weekly exercises.

Now, students can learn under the provision of feedback from PASS, and more exercises than before are given to students each week. Besides, the manual checking of programs by the tutor is now changed to automatic checking by PASS with a lot more test cases, allowing more thorough checking. PASS is responsible for the testing, and the instant feedback given to students provide them with informative clues about how to make corrections. It was observed that students were more motivated under this environment, and with extended supervision, there were fewer excuses to lag behind.

3 Effectiveness of Learning with PASS

Learning computer programming is best done by practice. As the tutor has made the weekly exercises with test cases available in PASS, students can work with the exercises anytime, anywhere. Besides, since instant feedback is available as students test their programs with PASS, they can make use of the feedback information to revise their programs and submit for testing again. In this way, learning to write programs is more effective within such an interactive and progressive learning cycle [4].

A number of similar automated programming assignment assessment systems have been described in the literature [5, 6]. Some of these systems emphasize on system performance and reducing manual work of the tutor and do not elaborate on the effectiveness of learning from the students' perspective [5]. Some other systems such as Online Judge [6] aims at challenging the users: the test cases are hidden and the users are only informed of whether their programs are correct or not. The system does not tell the user on which test cases the program works correctly and on which test cases the program fails. In contrast, PASS is expressly designed to help beginners to discover any problems in their programs. In the first pilot run of PASS, there was a response of "correct" or "wrong" for each execution of the program with respect to a test case. With a thoughtful design of test cases, we believe that each test case may help the students to pin-point to some possible logical faults in the program that cause the error. By adding annotations to each test case in the next release, the feedback returned to students will better help them understand their logical errors and thereby make the appropriate corrections.

During the semester, a tutor has closely observed two students in their laboratory sessions. Both students were slow learners and had no programming experience prior to taking the course. These two students worked on every laboratory exercise and used the feedback from PASS to revise their programs. Their progress was slow, but steady and encouraging. They found the availability of PASS suited their learning pace, and the testing environment provided by PASS provided them with useful feedback when revising their programs. Towards the end of semester, both students were able to write long programs on their own. We believe that this would not have been possible without an automated assessment system such as PASS.

PASS has been in operation for two semesters in four computer programming courses. A survey was conducted at the end of the first semester to evaluate the usefulness of this system from the perspective of students in two courses (the other two courses are still ongoing). The results are summarized as follows.

- The majority of the students reported that they used PASS about 5 times a week to check their lab exercises. On average, a student uses the system about 7 times a week, and a few students used the system more than 30 times a week.
- About 70% of the students reported that PASS did help them to reveal bugs.
- When asked to compare their understanding of lab exercises with PASS and without PASS, more than 80% of students considered PASS quite helpful or very helpful in their lab exercises, and about one out of four students requested to have PASS available for every program in the lab exercises. (We had intentionally left out some lab exercises which were not made available in PASS.)
- Students were asked to rate PASS on a scale of 1 to 5, and the distribution of their ratings is shown in Table 1. We note that about two-third of the students gave PASS a rating of 4 or above. It is encouraging to know that most students appreciated the assistance from PASS in their learning of computer programming.

Table 1. Distribution of students' rating of PASS (5 being the best rating)

Rating	1	2	3	4	5
Percentage (%)	3.0	8.9	24.4	50.4	13.3

- In their written comments, most students expressed their view that PASS was very helpful in facilitating their learning process. Below are some comments from students extracted from the survey (minimally grammatically edited):
  - I can work more independently and it gives me confidence when I got all correct. Little by little, I build up my own reliance!
  - Very good, the fast response can help me to follow up errors at once.
  - Build up my confidence.
  - Make the correctness of our programs higher, even without tutor's help.
  - Sometimes I thought my program was correct and actually it is incorrect and PASS helps me to know more about my program, so that I can revise again.
  - It's very useful because the test cases sometimes can notify me to think more carefully when coding.
  - It can help me to check my lab exercises by myself. It can encourage me to do all the lab exercises. So it is very useful.
  - We can know the bugs immediately; it increases the rate of learning.
  - It lets me know that my program still has bugs even after I tested it carefully.
  - I can check the answer and see any flaws in my algorithm or implementation.

These comments demonstrate that PASS has been well accepted by the students in assisting their learning in computer programming.

4 Conclusion and Further Work

This paper has reported the effective use of PASS in assisting the progressive learning of computer programming. PASS allows a tailor-made learning pace and style for students. It has provided a quick and convenient channel for students to test their work without manual involvement. Instant feedback to students encourages them to enhance their programming skills. Tutors observed that during laboratory sessions, some students simply used the system as a compiler and debugger without first test running the programs themselves, even when their program still had syntactic errors.

The introduction of PASS has made the learning of computer programming more rewarding than before. Both tutors and students are encouraged to witness the mobility and flexibility of learning supervised by PASS. We have used PASS in four programming courses with favourable student responses. In view of this, we plan to extend the use of PASS in more programming courses.

Currently, PASS performs the checking of outputs by simple textual comparison. Therefore, students must adhere to a fixed input and output format when writing their programs in order to satisfy the checking requirement. Working with such formatting constraints can be quite a hassle, as negligible differences between the outputs can result in the program being treated as "incorrect". Some students found such a restriction frustrating, and a number of them requested for higher flexibility. Presently, to circumvent this limitation, the tutor manually compared the outputs when such a claim is raised by students. We plan to improve this aspect of the system in the next release of the system. Moreover, we plan to attach annotation to each test case so that the feedback will become more informative and provide more concrete assistance for students to revise their programs, and debugging will become more interesting.

References

1. Woit, D., Mason, D.: Effectiveness of Online Assessment. SIGCSE Bulletin, Vol. 35.1 (2003) 137-141
2. Sheard, J., Dick, M., Markham, S., Macdonald, I., Walsh, M.: Cheating and Plagiarism - Perceptions and practices of first year IT students. Proc. 7th Annual Conference on Innovation and Technology in Computer Science Education, Denmark (2002) 183-187
3. Califf, M., Goodwin, M.: Testing Skills and Knowledge - Introducing a Laboratory Exam in CS1. SIGCSE Bulletin, Vol. 34.1 (2002) 217-221
4. Chong, S.L., Choy, M.: Towards a Progressive Learning Environment for Programming Courses. Int Conf of Web Learning: New Horizon in Web-based Learning, (2004) 200-205
5. Luck, M., Joy, M.: A Secure On-line Submission System. Software Practice and Experience, Vol. 29(8) (1999) 721-740
6. Kurnia, A, Lim, A., Cheang, B.: Online Judge. Computers & Education, Vol. 36(4) (2001) 299-315