INVESTIGATING PROGRAMMING FEEDBACK TO IMPRO\VE STUDENT LEARNING EXPERIENCE IN COMPUTER SCIENCE EDUCATION

Smart Learning Conference / Innovation Arabia 10

Programming is a core competence in computer science. Teaching programming is challenging due to the very nature of the skill itself i.e. programming is something that is best learnt over a long period of time and with practice. Therefore it is of at most importance to investigate whether the teaching and learning activities used in a programming class contribute to a positive learning experience.

Investigations:

1. Which **Teaching & Learning Activities** do learners on a computer programming course find support their learning: do teaching staff concur?
2. What are the factors the make **auto graders** help computer science students to have a positive learning experience?
3. Does incorporating **peer-feedback by peer-testing** of programming artifacts promotes deep learning in CS programming courses?

Teaching & Leaning Approaches

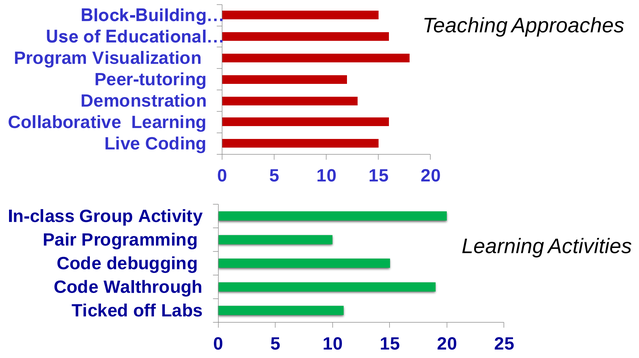
Which Teaching & Learning Activities do learners on a computer programming course find support their learning: do teaching staff concur?

Research method:

The participants of this study were 20 students from second-year B.Sc computer systems, and 4 faculty members of the School of Mathematical and Computer Sciences of Heriot Watt University, Dubai campus. Questionnaire was designed and distributed to the students and One-to –one interview was done with the faculty members. Creswell's (2009) data analysis steps were used to analyze and organize the data collected in the study to generate the research findings.

Results:

This research reveals that the learners have varying conceptions about what it means to program. It is evident that to acquire the skills of program design and problem-solving, hands-on and a practical oriented approach is essential.



The teaching of programming in Computer Science courses partly relies on coursework for which students are expected to program a piece of software matching a given specification. Students are expected to test their programming artifacts to verify its correct behaviour. Students receive feedback on their programming usually in lab sessions, in demonstration sessions or as part of their coursework feedback .This study reveals that staff and student agree that constructive & personalized feedback play a vital role in the learning process.

Auto Graders

What are the factors the make auto graders help computer science students to have a positive learning experience?

Inquiry:

Evaluate the possibility of implementing auto graders in the first year introductory programming as a potential learning tool for formative assessments.

Research method:

Auto grading software was initially demonstrated to the students. Year 2 BSc Computer Systems students where chosen for this study. The students were then asked to complete an online survey questionnaire handed via their class representative. A total number of 18 complete responses were received. An interview session conducted with 4 staff members of the department. Data analysis methods adopted from Creswell's six steps of analysis to ensure improved organization of data.

Results:

Lecturers perceive auto grading would be useful for conducting formative assessment in introductory programming courses. Students liked getting immediate feedback, they perceived auto graders as a positive learning aid. Lecturers believe auto graders can save time in assessing lab work. They also consider the system to be fair as evaluation criteria are the same for all. Both students and staff considered that the human element/connection is always required for staff-student relationship.

Peer-Testing as Peer-Feedback

Does incorporating peer-feedback by peer-testing of programming artifacts promotes deep learning in CS programming courses?

Inquiry:

Feedback on programming coursework takes time for the teacher to produce, as a result comes late after the student have submitted their work. Peer-feedback by peer-testing could shorten feedback time, could promote critical analysis of programs through practice of testing and review.

Research method:

Two perspectives investigated: the teachers' point of view, and the learners' point of view. Qualitative and empirical methods were used combining interviews with teachers on programming feedback, and small scale peer-testing experiment followed by a focus group discussion with Year 3 BSc Computer Science students.

Results:

On programming feedback, teachers stressed the importance of not giving the answer straightaway, of giving self-analysis guiding, of using tests as mean to mark and prepare feedback. On peer-testing experience, students highlighted the benefits of seeing other ways of doing, and of having someone looking at own program.

Conclusion and Future Works

Conclusions:

These inquiries showed the importance of varying the teaching activities and of offering multiple forms of feedback and programming practices. Students highlighted both the **added value of automatic learning aid** as well the **importance of human feedback**.

Future Works:

An ongoing research funded by Heriot-Watt's *QAA Learning and Teaching Enhancement* is investigating how to help **student transition from passive learner to critical evaluator through peer-testing of programming artefacts**.

The proposition is to organise software testing of students' programming artifacts by the students themselves. Our hypothesis are as follows. (1) Students are encouraged during the peer-testing to **engage** with the coursework in a manner different from the traditional problem-solving way. (2) Students get to comprehend and review someone else's programming code which promotes **critical analysis** of their own programming. (3) Students **learn from their peers** by receiving feedback in the form of tests and comments. (4) The students **focus** on the coursework does not decline between submission and receiving feedback as they are participating in the testing activity.

References

Brown, G.A., Bull, J. and Pendlebury, M., 2013. *Assessing student learning in higher education.* Routledge.

Falchikov, N. 2013. *Improving Assessment Through Student Involvement: Practical Solutions for Aiding Learning in Higher and Further Education*. Routledge.

Isomöttönen, V. & Tirronen, V., 2013. *Teaching Programming by Emphasizing Self-direction: How Did Students React to the Active Role Required of Them?* Trans. Comput. Educ., 13(2).

McAllister, G. and Alexander, S. 2009. *Key aspects of teaching and learning in computing science.* Handbook for Teaching and Learning in Higher Education: Enhancing academic practice.

Miller, C.S. & Settle, A., 2011. *When Practice Doesn't Make Perfect: Effects of Task Goals on Learning Computing Concepts.* Trans. Comput. Educ., 11(4).

Sorva, J. 2013. *Notional Machines and Introductory Programming Education.* Trans. Comput. Educ., 13 (2).

Smitha Kumar (Dubai)
Manuel Maarek (Edinburgh)
Talal Shaikh (Dubai)

School of Mathematical & Computer Sciences
Dubai and Edinburgh campuses
Heriot-Watt University