Report on COMP 1730, Semester 2, 2015

Course Title. Programming for Scientists

Convenor. Patrik Haslum

Number of Students. 309 (COMP 1730) plus 20 (COMP 6730)

History. This course has had severe problems in the past (overall satisfaction in SELS at 20% in 2010 and 2011) but has been turned around more recently (satisfaction over 60% in 2013 and 2014). In 2015 the percentage for the satisfaction question was down at 51% but the average score for that question remains at 3.4 which is in line with the 3.5 and 3.6 for the previous two years. Hence part of the overall picture is that a big improvement has been made over the last 4 years and needs to be continued rather than radically changed. Note also that the numbers of students taking this course have increased markedly (roughly doubled since 2012, and before that it was a small course).

Suitability of Prerequisites. No prerequisites. The convenor (Patrik Haslum) notes that a small number of students are too advanced for the course and there were enrollments of 3rd year CS students.

Composition of Student Cohort. Of the 100 students who gave feedback, 66% were domestic. 68% are first year students with 58% reporting no prior programming experience. More than half of the students were enrolled in Engineering (excluding SE) and about 40% were Science students. The rest are scattered across schools.

Relevance of the Course for Students. The course is a general introduction to programming for non-CS students. It is required for 2/3 of students who provided SELT feedback.

Suitability of Learning Offerings and Assessment. Assessment was by means of two assignments at 15% each, two exams (mid-semester at 20% and final at 40%) and five lab quizzes (total of 10%). As many students noted in the open-ended SELT feedback and the evaluation done by the convenor, it was felt that there was an imbalance between assignments (the 2nd was considered too difficult and less relevant). Learning offerings are good but one may consider more “hands-on” ways of delivering content.

Feedback and Availability of Resources. Availability of Resources was good, although some students felt that the second assignment should have been released earlier in full.

Percentage of Student Feedback. 100 students gave feedback out of a total of 309 students enrolled in COMP 1730.
Recommendations. The course faces the same difficulties as any other introductory programming course. The main challenge is to create a level playing field, as roughly half of the students have prior programming experience, as well as dealing with a student cohort that is diverse both in terms of cultural and educational background. The course was subject to an external review conducted by Bernd Meyer (Monash) and the suggestions provided in the review were carefully implemented by the convenor. The convenor’s report provided for this review thoughtfully analyses and evaluates and reflects on the course and echoes some of the recommendations given below. One difficulty noted by the convenor was plagiarism.

1. Balance the difficulty of the two assignments and make sure that they are fully released in a timely fashion.

2. As suggested in Bernd Meyer’s report, consider streaming of students. This could be done by offering practical hands-on sessions that are voluntary, maybe through PAL, or by simply providing drop-in sessions where students can bring laptops and ask questions, or an extra tutorial for beginners. It may be worth while to couple this giving students feedback into the course by means of (weekly?) polls (e.g. “how confident do you feel to write a recursive program?”) to inform content of a possible extra tutorial for beginners, and also to direct students (e.g. “if you have answered yes to any of the above, we strongly encourage you to attend”).

3. As suggested by the convenor, students that are clearly too advanced for the course should not be enrolled.

4. Specialise the generic policy on plagiarism to the special case of CS. Elaborate more on plagiarism in the administrative course handout, possibly in the style of frequently asked questions. What is OK and what is not OK? For example, is it OK to discuss assignments with peers? (I would say yes). Is it OK to scribble on paper? (I would say yes). Is it OK that every participant of the discussion receives a copy of the notes (I would say no – they should be discarded). Is it OK to sit down in a group in front of a computer and discuss the assignment? (I would say no). Is it OK to share files related to the assignment (Of course not!). What happens if someone copies my solution (I will loose marks due to collusion). This list is clearly not exhaustive and may seem to detailed but a specialisation of the generic policy on plagiarism for the concrete case of programming courses may be appropriate as more than half of the cohort are non-CS students.

5. Consider an even more flipped classroom model. The convenor has implemented a flipped classroom model as suggested by the external course review. As noted in the convenor’s comments, this changed the
style, but not the content, of student’s comments in the evaluation. Compared to previous editions, the amount of live programming has increased but students still feel that they learn more by programming themselves compared to watching someone else. One may consider to dedicate, say, one lecture slot, to group programming exercises and shifting the background material to reading assignments.

6. In the individual SELT rubrics, the course was rated lowest for feedback (3.3). The convenor should explore ways to provide more student feedback, ideally on a weekly basis, e.g. through lab quizzes. Also, can the self-test questions contained in the textbook made electronic and can one require that all students complete them on a weekly basis?