The problem...

Many computer science modules build on skills and competencies which students are assumed to have acquired in introductory modules. However, students can enter modules by a variety of routes; many lack confidence with what are seen as basic programming skills, and find it difficult to reflect on their exact levels of skill in the many component areas that make up deep knowledge of software engineering. This can reduce their persistence on important challenges, and seriously impact on achievement.

In the module described here, common problems had historically been:

- An unwillingness to debug code, leading to poor performance on coursework.
- Problems with generalising specific lab challenges to unfamiliar problems.
- Anxieties about independent work (and hence high levels of plagiarism).
- Unwillingness to use unfamiliar technical tools such as Javascript debuggers.

The students... aka the solution.

This module (Web Development) was a core second-year requirement, but was also taken as an option by third-year students from many other courses. Past experience of practical programming was highly variable, and this led many students to assume that others in the class were more expert than them, and to be anxious and concerned about this.

Most students found it difficult. Many because the module was focused upon GCSE students; it was "unreliable as the 'learning' tool because most GCSE students are set to struggle or fail in this way of teaching. The end of the list should be that students are particularly know-how to test and evaluate the software's ability to meet their needs. Please, please, please, do not offer this module to first-year CE students if you are not willing to go to this length, because it may be tedious to the CE students; it is necessary to order on this basis and not at all if you are offering this module to CE students. Separate lab sessions should be set for CE students and the weighting of coursework I would claim should be changed to suit CE students if you would like to fix this module for both CE and CoT students.

Students varied in the degree of help they wanted, but all believed they were capable of 'fixing the problem' themselves.

Students varied in the degree of help they wanted, but all believed they were capable of ‘fixing the problem’ themselves. However, few found it easy. Students who fixed with progressively less instruction.

Extra labs and tutorials, to (a) demonstrate that problems can be tracked and fixed (b) make the use of debugging tools routine and familiar. Students taking the module: 76

Hits on the discussion board: 2989

Messages posted: 78

Students who posted : 19 (25%) Questions answered by another student: 20%

Only 4 plagiarism / collusion cases *
(as opposed to 15 in the previous year)

What next?

Encourage students to post their own videos, to show they've fixed problems being fixed "while they're doing one to one". Measure self-efficacy using a standardised scale [3], to permit meta-analysis and longitudinal study. Continue to design labs which provide many and early opportunities for success. Encourage students to reflect on how they physically respond to uncertainty and novel challenges, and whether this affects their impression of their own ability.

References / inspirations


EVALUATING THE OUTCOMES: 1. Increased confidence across a broad range of topics

<table>
<thead>
<tr>
<th>Key:</th>
<th>1 = Very confident</th>
<th>2 = Confident</th>
<th>3 = Quite confident</th>
<th>4 = Not at all confident</th>
<th>Start of module Min – max (mean) **</th>
<th>End of module Min – max (mean)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How the Web works – theory (e.g., protocols and languages)</td>
<td>2-4 (2.76)</td>
<td>Covered in lectures and labs as a matter of course.</td>
<td>2-4 (2.90)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Working with the DOM (e.g., updating paragraph elements in a document)</td>
<td>1-4 (1.84)</td>
<td>All students had in theory already passed this topic, but knowledge was highly variable: The first 3 labs were adapted to revise the topic, and re-use previous year’s materials.</td>
<td>2-4 (2.41)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Debugging java (e.g., identifying a problem within a ‘while’ loop)</td>
<td>1-3 (2.23)</td>
<td>This should be a basic skill for students at this stage. Practice exercises were introduced, e.g. code with deliberate bugs which students fixed with progressively less instruction.</td>
<td>2-4 (2.97)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Debugging javascript (e.g., identifying a problem stemming from a null var)</td>
<td>1-3 (2.08)</td>
<td>An unfamiliar activity which requires the use of a novel tool (Firefox). Deliberate errors were introduced into lab example code for students to track down. All but 2 labs explicitly required Firefox to be used, and several video tutorials demonstrated its use.</td>
<td>1-4 (2.50)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Breaking code to see the effects (e.g., overflowing a java Array or replacing null variables)</td>
<td>1-4 (2.16)</td>
<td>I wanted to encourage students to ‘play’ and experiment without fear that their work would be destroyed. Lab demonstrators showed how to temporarily, and safely, generate error messages. Lab videos showed many common code errors and fixes.</td>
<td>2-4 (2.77)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Client-server interactions (e.g., POST form responses)</td>
<td>1-4 (2.05)</td>
<td>Covered in lectures and labs as a matter of course.</td>
<td>2-4 (2.50)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML syntax (e.g., rules to which XHTML documents conform)</td>
<td>1-3 (2.05)</td>
<td>An intimidating topic for many students, covering a broad range of topics. Extra labs and tutorials were produced, with model answers and examples.</td>
<td>2-4 (3.00)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. Independent working

4. Reflection on skills and needs

43% of students felt that they were more general than what they had learnt out of its specific context.

5. Better achievement on practical assignments

Between 2009 and 2010, the average module mark increased from 48.5 to 58.6. 60% of this rise was due to increased marks on practical coursework. Exam performance also improved, particularly on questions which assessed experience or critical analysis. While the range of coursework marks in both years ran from 0 to 100%, the 23rd percentile shifted upwards from 15 to 51%.

Students varied in the degree of help they wanted, but all believed they were capable of ‘fixing the problem’ themselves. However, few found it easy. Students who fixed with progressively less instruction.

** Survey statistics for 2010 are based on 66 students (87% of cohort).