COMPUTER SCIENCE, SCHOOL OF MATHEMATICAL & COMPUTER SCIENCE

Transition of automated feedback into student's learning environment

Graeme Reid, Gudmund Grov, Verena Rieser, Michael Lones

INTRODUCTION

Programming is a key skill for a computer scientist

Many Computer Science (CS) students struggle with programming

Studies have shown that constrained lab times and insufficient feedback is an issue

AIM

To develop technology that provides automated feedback which will help first year CS students to become independent learners

We wished to

Liberate staff from low-level and repetitive questions enabling them to focus on deepe learning problems.

Provide **automatic feedback** and students could make meaningful progress inside and outside lab hours.

Enhance existing tools with feedback so students do not need to learn yet another tool.

OUR APPROACH

Analysed existing labs to understand which parts could potentially provide automated feedback.

Developed a plug-in for the Eclipse tool (which is used in the labs) based around code analysis and testing that could provide feedback for one lab.

Applied the tool in a lab and asked students to fill in a questionnaire.

Analysed results and improved our tool based on it.

INITIAL DEVELOPMET & ANALYSIS

An initial tool was developed and applied in a lab.

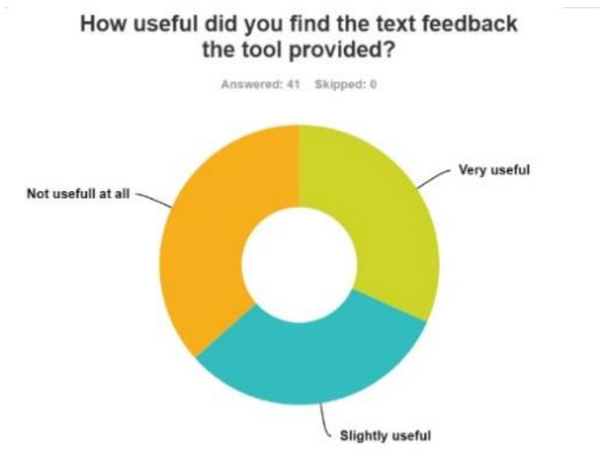
Sample lab questions

Recursive sum of numbers
Write a recursive method ***sum****, which given a number n, returns the sum of all positive numbers up to n - that is, it computes:*
sum(n) = 1 + 2 + ... + n

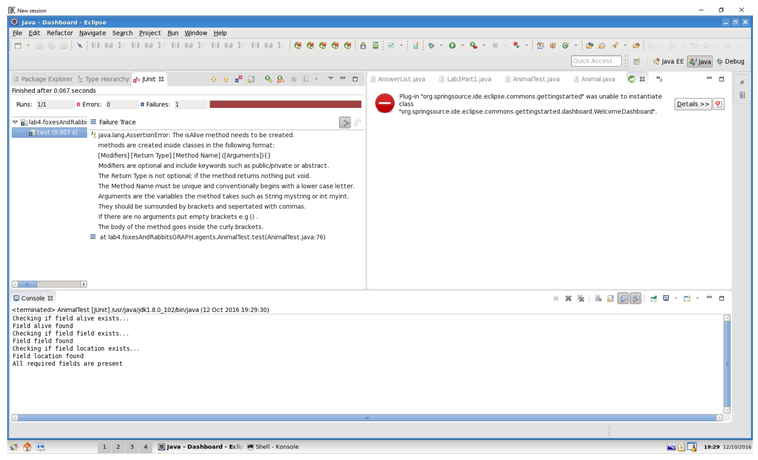
Recursive multiplication using only addition
*Write a recursive method multiply which given two (positive) integers m and n as arguments, computes m*n using only addition (and recursion).*

Computes a Fibonacci number
The Fibonacci sequence is 0,1,1,2,3,5,8,13,21,34,.... You should write a method Fibonacci which given n as an argument, returns the nth Fibonacci number using recursion, that is Fibonacci(0) = 0, FiboNacci(1) = 1, FiboNacci(2) = 1, FiboNacci(3) = 2, FiboNacci(4) = 4, FiboNacci(5) = 5, FiboNacci(6) = 8, FiboNacci(7) = 13, FiboNacci(8) = 21, FiboNacci(9) = 34 and so on.

Main result: Students liked it and value its potential but struggled to understand the feedback as it was too verbose and generic.



Previous method of providing feedback



Having the feedback appear in multiple places was not ideal

The console display could be shrunk, moved or closed by users of Eclipse, meaning the size and location of the feedback could vary.

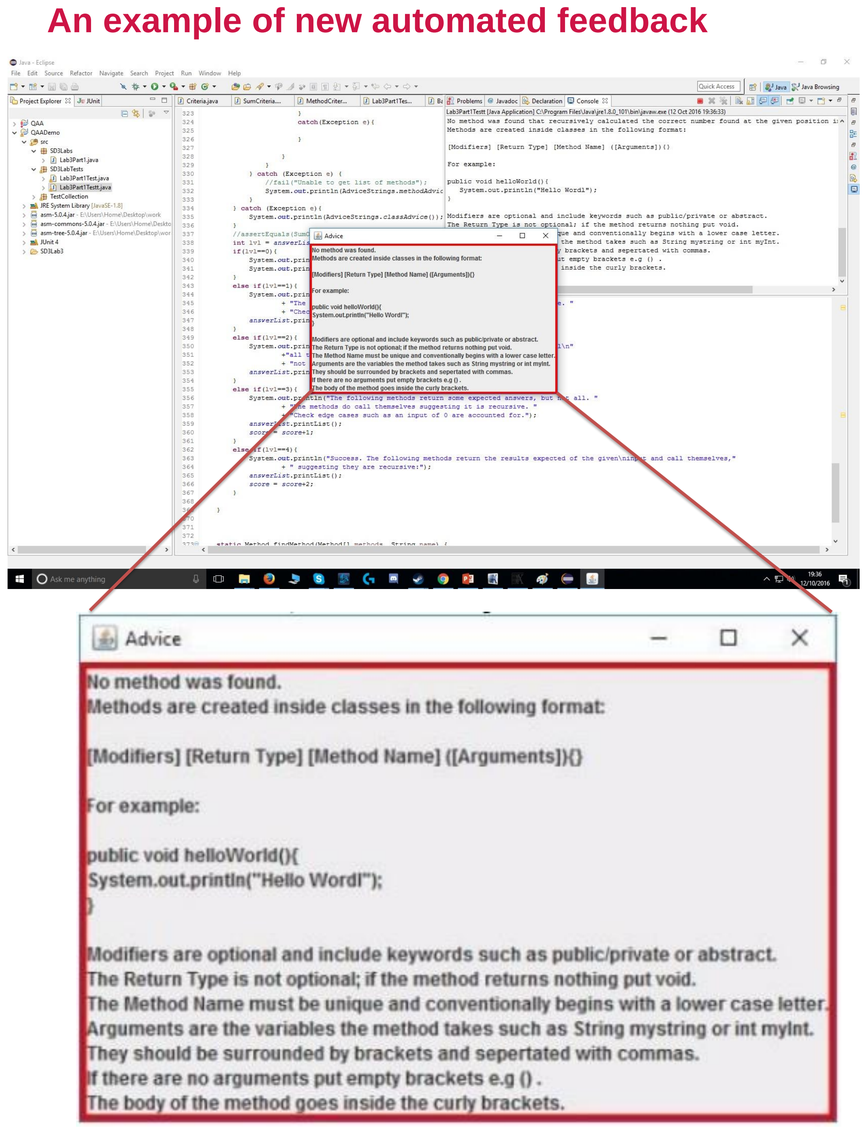
Requirements for updated tool

Use pop-up windows to deliver targeted advice closer to code.

Pop-up window should be able to be repositioned and kept open for reference as the student continues to work.

AN UPDATED FEEDBACK TOOL

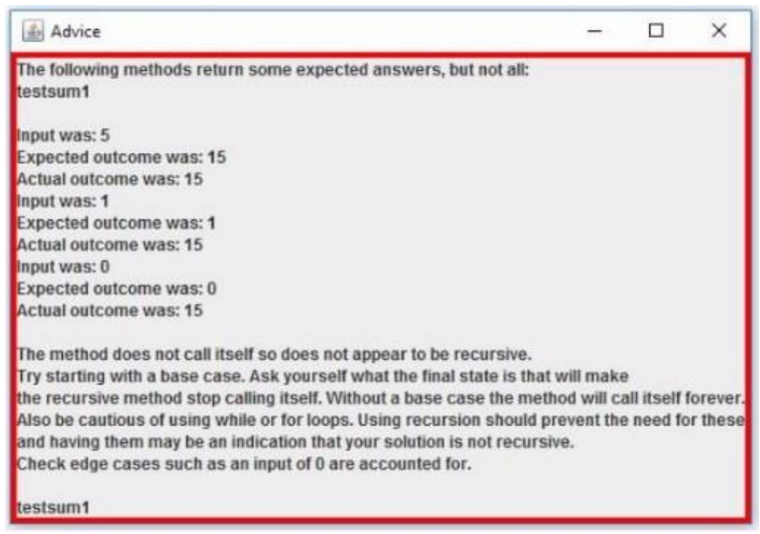
An example of new automated feedback



In the new version the advice is more tailored to specific circumstances and shown as pop-up closer to the problem.

A feature requested in the feedback was to show the answers that the test was expecting and the answers that it had received from the output of the students program.

Feedback with expected & actual results



ONGOING & FURTHER WORK

Combine labs and tests to automatically provide feedback for more complex problems

Re-factor the code to make it easier to add create, combine and configure new labs and tests.

Investigate how the work can be extended to automated marking.

BIBLIOGRAPHY

1. Michael McCracken, Vicki Almstrum, et al. . A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. In Working group reports from ITiCSE on Innovation and technology in computer science education. Pages 125-180, ACM, 2001.

2. Arto Vihavainen, Thomas Vikberg, Matti Luukkainen, and Martin Pärtel. Scaffolding students' learning using test my code. In ITiCSE '13, pages 117-122, ACM.

3. Diane Litman. Enhancing the Effectiveness of Spoken Dialogue for STEM Education. In SLaTE, pages. 13-14, 2013.

HERIOT WATT UNIVERSITY

QAA