Tutorial 3: More fun with Equations — testing and fixing

In the last tutorial you searched for faults in the “Equations” program. In this tutorial, you will diagnose and fix some of those faults.

Exercise 0 – Catching up

If you haven’t completed at least the exercises from last week up to and including exercise 3 (“Test if the SUT meets your requirements”), then do so now.

Exercise 1 — Describe some failures

New Challenge: describe test failures in a way that others can understand them and judge their significance

1. Make a Word (or similar) document containing a form (as a single table) for recording test results. Think about what information you need to include.
2. Populate the form with at least three of the failures you have observed
3. Show the form to a TA for feedback

Exercise 2 — Diagnose some faults

New Challenge: diagnose the reason for a failure you’ve found

For at least one of the failures you described in the previous exercise, work out what is causing it — what the fault in the code is that gives rise to it.

Exercise 3 — Use tests as specifications

New Challenge: define additional tests to tightly specify a needed behaviour

For at least one of the faults you diagnosed in the previous exercise, expand your tests

In particular, if you found the fault using manual tests, try to write some automated tests that capture the behaviour. You may need to do this at low level e.g. unit or integration.

Don’t expect all of these tests to fail when faced with the faulty code. It’s quite possible that the current code has *part* of the functionality you want.

Exercise 4 — Change the code so that your tests pass

New Challenge: Make the smallest possible change to a program such that it passes relevant tests

For at least one of the faults you have specified using tests, fix it.

Make sure that in the process you don't break any other program behaviour. You can, of course, use your existing test set as a regression test suite for this.

Exercise 5 — Use exploratory testing (EXTENSION)

New Challenge: Perform principled exploratory testing

If you haven't already, apply a principled exploratory testing approach to find additional faults.

1. Define the principles by which you will test in this case (see the example in lecture 2)
2. Test Equations using those principles

Exercise 6 — Cross-check your results with someone else (EXTENSION)

Once you've fixed a number of faults, ask another student from the class to run their tests against your version of the code. Do they find only the same things as you do?