Guidance for revision for Ant Junit lab test for Comp220 and Comp285.

The test will be very similar to last year's test which has been posted online.

Conduct of the test

The test is an open book test, you can use any files you used and created in the lab session.

The test can be done at home on your own computer or in the lab.

The submissions will be plagiarism checked.

This test is a time limited test, you will have 4 hours.

Example Ant build

As part of this revision guide there is an example directory.

You **CAN** use this example build file to get your started.

Instructions are on Canvas on how to get Ant and Junit working on both Windows and MAC computers. The example build file works for both Windows and MAC.

Getting started

The example build file already does the following;

Compiles the target and test code.

Runs the Junit tests.

(It also has some mutation testing code to help you to develop the tests, see below).

It does **NOT**

Produce XML and HTML reports (to do this you will need add in the Ant tasks, see lecture_ant_testing.pptx and lecture_ant_testing2.pptx). You also need to add in properties and comments.

Mutation testing

To test your tests, the following procedure is a good start.

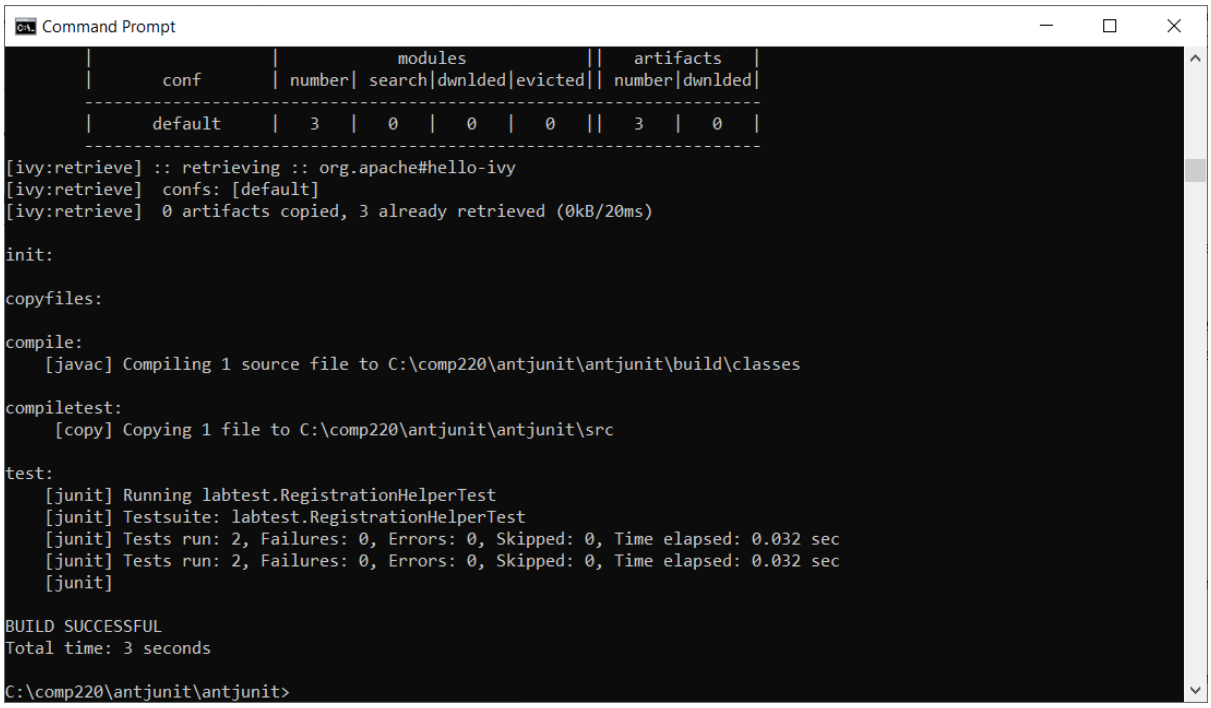
1) Introduce bugs to the target source code
2) Add or modify tests to reveal the bugs
3) Run the tests
4) If the test passes, go make to step 2, and add more tests or modify your tests

This process of "testing tests" is called mutation testing. Each source code with bugs in it is called a source mutation. In the build I have given you there are a number of mutations, called mutation1, mutation2, mutation3 etc. The name is the name of the directory containing the mutation. Have a look at the mutation source code, so you can see the bugs. Add in more mutations to complete the task.

To run the tests without the mutation, do the following:

ant test

The output should look something like the following.



Notice no failures, no errors. This is correct.

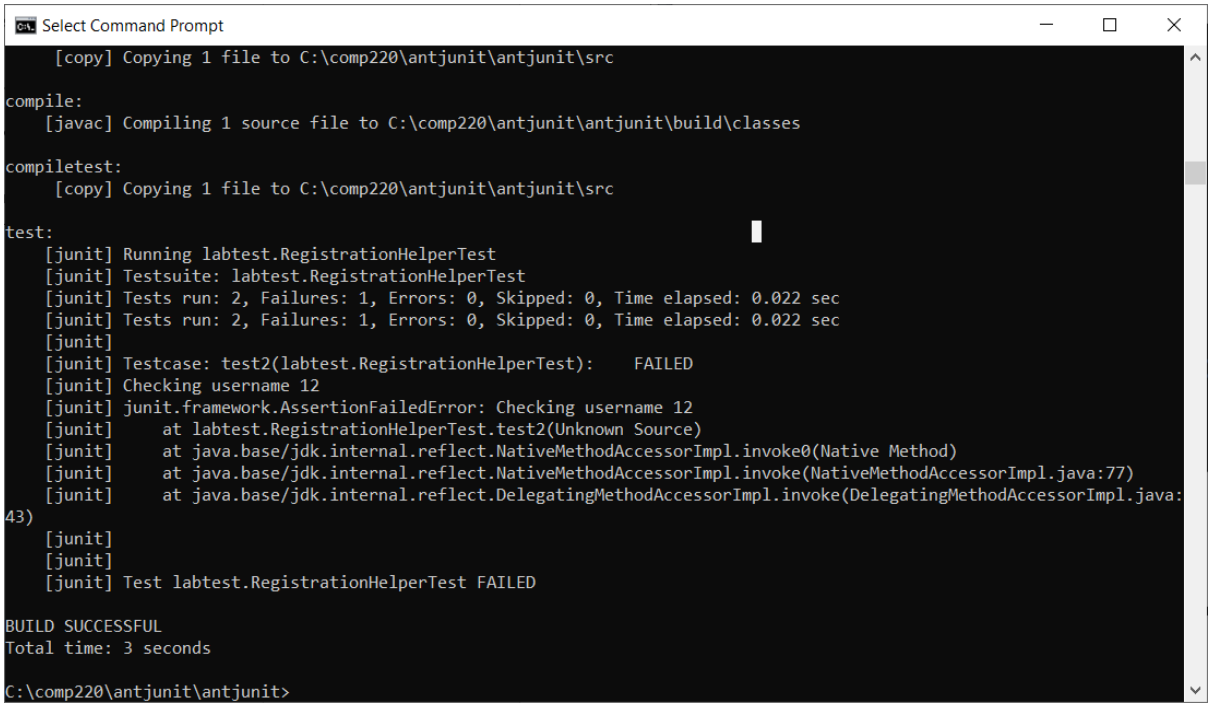
If when you run the code without the mutation, and you get a test failure, then something has gone badly wrong as the tests with the working code should all pass. If you get a failure with the working code, change your tests until they pass.

To run the tests with the mutation, pass in the Ant property mutation, for example to run the code with mutation1, type the following.

ant -Dmutation=mutation1 test

Using the -D option to set the property.

The output should be something like this.



Notice at least one of the assertions has failed.

For the example code given, I've provided 4 mutations, but you should add more. For mutations 1 and 2, I have added in tests for these mutations, so the tests will fail. For mutations 3 and 4 you should add tests so the testing fails.

So to recap the rules are as follows.

Whenever running the working code, the tests should ALWAYS pass.

Whenever running the buggy (mutation code), the tests should ALWAYS fail.

So, your task for this revision is:

Update the build and Java file as requested by the PDF.

Fix the testing for mutations 3 and 4 by adding in more assertions.

Add in some more mutations and for each one of these fix the testing.