Tutorial 2

**Objective**

During this week’s tutorial you will learn to create test data for the unit testing of input forms.

The aim of the tutorial is to help you:

1. Write test cases that are detailed enough to be correctly conducted by other people.
2. Develop test data that can be used for testing that provides a balance between comprehensiveness and the time and cost taken to conduct a test
3. Understand how to create test data for a system input by analysis of the significantly different options for a variable.

**Required resources**

To participate in this tutorial:

- A pen and a notepad

**Note**

Solutions to all the exercises are available on-line on the unit web-site. Please don’t look at them until you are finished unless you are stuck or need clarification. On-campus students - there are a few more exercises here than you will probably get to finish in class. Don’t worry about that you can simply finish them in your own time after class.

**Background**

The lecture this week covers software testing. One critical skill you require to conduct good software testing is the ability to create good test data. This tutorial will provide you with some exercises to help you learn how to do this. Tests are often (and best) conducted by people other than the original developers of the software. It is important that test cases and test data is specified in enough detail for them to be able to be clearly understood and repeated.
Consider the process used to login and logout from the my.monash web site. How would you test that function? At a high level the test process might be described as:

1. Login to the site my.monash.edu.au with a valid username and password.  
   Expected result: The my.monash.edu.au homepage is displayed.

2. Click the logout link on the my.monash.edu.au homepage.  
   Expected result: the my.monash login page is displayed.

That high level process is nice for understanding a bit about the nature of the test and the “thing” that is being tested, but it isn’t good enough to conduct the test. More detail is needed to make sure that the test, when it’s conducted, is unambiguously repeatable. A better amount of detail is illustrated by the test case that follows (note the table format, and also the columns that allow a quick indication to be made of whether or not the expected results were found, and for the tester to make a comment.)

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Expected result</th>
<th>Passed</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Open the URL my.monash.edu.au</td>
<td>The login page is displayed. Verify that the input fields username and password are displayed. Verify that the login button is displayed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Once login page is displayed enter username and password. Enter “pwd123” into username field. Enter “abc123” into password field.</td>
<td>The my.monash home page should be displayed. Verify that the page displays a personalized “Welcome Peter” message in the center of the page header. Verify that the page displays a “logout” link at the top right side of the page header. Verify that the user has successfully logged out of the application and the my.monash.edu.au login screen is displayed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Click the submit button.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Click the logout link at the top right side of the page header.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Note how this (more detailed) test case could be repeated by any person - whether or not they knew anything about the my.monash web site. Also note, that the same process could be repeated with different input data. In the example above the data entered is valid, a similar process could be created for data input that is invalid.

### Tasks (on-campus students)

1. Developing a test case 1. Use the example given above (the my.monash login page) as the basis of a test case for the entry of an invalid username and password. (Create that test case using the same table format used above).

2. Developing a test case 2. Have a look at the example form located at [http://www.wufoo.com/gallery/templates/forms/contact-form/](http://www.wufoo.com/gallery/templates/forms/contact-form/). (Wufoo is an on-line form generator, This is one of the example forms that they have to show different ways their technology can be used.) If you don’t have a web connection the form is shown below. This form might be used on a web site to allow visitors to the site to ask a question or make a comment. The message field is mandatory. The email address is optional. Write a set of test cases to test this form.

   ![Contact Form](wufoo.png)

   2.1. Once you have finished, compare your test cases with those developed by another class member.

3. Creating test data. They key to developing a good set of test cases is to develop good test data. To do a good job of testing, the test data developed should cover all the significantly different possible inputs. Testing the same type of input again and again doesn’t help gain confidence that a system is working as expected, but testing significantly different options does. For example, if testing a text input for a last name: typing in “Smith”, “Brown”, “Jones” as different options doesn’t add much to the quality of testing as they aren’t significantly different. However, “Smith” and “O’Donnell” are different (“O’Donnell” is different because it contains a valid but special character - the ‘ ’ - you wouldn’t believe the number of systems that won’t accept names with these special characters) and are excellent options to include in the test data set.

   Now, imagine an input field that should only process an identification number that has 9 numerals. Create all the significantly different options you think are needed to test that input. In your set, try to include values that are valid, invalid, near the boundary of valid and invalid and some values that are totally weird.
3.1. Again, once you have finished discuss your solution to that problem with a partner, then with your tutor moderating, with the rest of the class. How many options did you come up with? How many did the class come up with?

4. Designing for input errors. Often the designers and programmers can limit the possibility of input errors by the way input screens are designed. This is part of another example form from Wufoo (http://www.wufoo.com/gallery/templates/forms/emergency-contact-and-medical-info/).

What techniques has the form designer used to make sure that the users can’t input incorrect data?

5. Testing different combinations of hardware and software. Imagine you have been hired to test a web-based software application. The owners of the web site want to make sure that the application works on all the major operating systems, browsers, on desktop computers, laptops, tablets and smartphones. How many different combinations of these variables can you think of (each combination represents an environment that the application needs to be tested against)? If the test suite that has been developed take 25 minutes to run, how long will the complete set of tests take? If you get paid $55 an hour, how much will the tests cost to run? What other costs will be associated with the testing (other than your labour costs)?

6. Look at the following Scribble code. The code is supposed to make a sprite turn around in a circle after the green flag has been clicked. This code isn’t working. Outline a test plan for this script. Can you by examining the code, and thinking about what you know about Scribble work out the problem?

If you are able to work it out - sit quietly for a moment. If you can’t work it out after a few minutes ... find a partner who also hasn’t yet worked it out. Discuss it with them. Work through the program logic out loud with your partner. See if you can work it out.

6.1. Your tutor will lead a discussion about this. Was it easier for you (or for most people in your class) to find the solution to this problem on your own, or with a partner? Why?
7. That’s it - with about 15 minutes to go in the class your tutor will distribute the first in-class test for the semester. This is worth 1% of your mark for the unit. It should only take 5 to 10 minutes to complete. You must work on it on your own. Give your answer sheet back to your tutor at the end of the class. You will get the result next week (don’t worry, it’s not hard).

Tasks (off-campus students)

1. Developing a test case 1. Use the example given above (the my.monash login page) as the basis of a test case for the entry of an invalid username and password.

2. Developing a test case 2. Have a look at the example form located at http://www.wufoo.com/gallery/templates/forms/contact-form/. (Wufoo is an on-line form generator, this is one of the example forms that they have to show different ways their technology can be used.) If you don’t have a web connection the form is shown below. This form might be used on a web site to allow visitors to the site to ask a question or make a comment. The message field is mandatory. The email address is optional. Write a set of test cases to test this form. Once you have finished, post your solutions in the discussion forum on the unit web-site (to the forum just for off-campus students). Check back later after a few students have posted their solutions and compare your solution to theirs. Post any comments or questions you have about any differences between the solutions posted there.

3. Creating test data. They key to developing a good set of test cases is to develop good test data. To do a good job of testing, the test data developed should cover all the significantly different possible inputs. Testing the same type of input again and again doesn’t help gain confidence in a system, but testing significantly different options does. For example, if testing a text input for a last name typing in “Smith”, “Brown”, “Jones” as different options doesn’t add much to the quality of testing as they aren’t significantly different. However, “Smith” and “O’Donnell” are different (“O’Donnell” is different because is contains a valid but special character - the ’ - you wouldn’t believe the number of systems that won’t accept names with these special characters) and are excellent options to include in the test data set.

Now, imagine an input field that should only process an identification number that has 9 numerals. Create all the significantly different options you can to test that input. In your set,
try to include values that are valid, invalid, near the boundary of valid and invalid and values that are totally weird.

3.1. Again, once you have finished post your solution to the unit discussion forum (for off-campus students). How many options did you you come up with? How many did other members of the class come up with?

4. Designing for input errors. Often the designers and programmers can limit the possibility of input errors by the way input screens are designed. This is part of another example form from Wufoo (http://www.wufoo.com/gallery/templates/forms/emergency-contact-and-medical-info/).

![Emergency Contact and Medical Information Form](image)

What techniques has the form designer used to make sure that the users can’t input incorrect data?

5. Testing different combinations of hardware and software. Imagine you have been hired to test a web-based software application. The owners of the web site want to make sure that the application works on all the major operating systems, browsers, on desktop computers, laptops, tablets and smartphones. How many different combinations of these variables can you think of (each combination represents an environment that the application needs to be tested against)? If the test suite that has been developed take 25 minutes to run, how long will the complete set of tests take? If you get paid $55 an hour, how much will the tests cost to run? What other costs will be associated with the testing (other than your labour costs)?

6. Look at the following Scribble code. The code is supposed to make a sprite turn around in a circle after the green flag has been clicked. This code isn’t working. Outline a test plan for this script. Can you by examining the code, and thinking about what you know about Scribble work out the problem?

![Scribble Code Example](image)

If you are able to work it out - sit quietly for a moment. If you can’t work it out after a few minutes ... find a a friend (anybody who will listen to you for a few minutes). Discuss it with them. Work through the program logic out loud with your friend. See if you can work it out.
6.1. Was it easier for you to find the solution to this problem on your own, or with a friend? Why?

7. That’s it. On the “study guide” tab of the unit-web site for this week you will find a link to the first on-line test for the semester. This is worth 1% of your mark for the unit. It should only take 5 to 10 minutes to complete. You must work on it on your own. The test will be available for you to do for 1 week (you won’t be able to access it after midnight on Sunday). If you have any problem accessing the test please send Peter O’Donnell (peter.odonnell@monash.edu) an email.