University of Windsor
School of Computer Science
https://cs.uwindsor.ca

Master of Applied Computing
COMP-8117
Advanced Software Engineering Topics

Dr. Aznam YACOUB – https://www.y-az.one
519-253-3000 (ext. 3781) - aznam@uwindsor.ca
Office Hours : Thursday & Friday (9.00am – 12.00pm EST)

Lab 4 : Tests (1%)

Goal

In this lab, you'll learn how to test plan and implement them using testing tools.

Training (Week 1 – Not evaluated)

Go to this page : https://softwaretestingfundamentals.com/software-testing-exercises/

You will do the exercise 2 to 7 (0.5 pt per exercise). For exercise 5 to 7, you have to imagine test cases and write the tests specifications using the IEEE Standard (see notes below).

IEEE 829 - Test Design Specification

Creating the test design is the first stage in developing the tests for a software testing project. It records what needs to be tested, and is derived from the documents that come into the testing stage, such as requirements and designs. It records which features of a test item are to be tested, and how a successful test of these features would be recognized.

As an example lets use a Billing project from which the following testing requirements may be defined:

- A normal bill can be produced.
- A final bill can be produced.
- The volume discount is properly calculated.

The test design does not record the values to be entered for a test, but describes the requirements for defining those values. This document is very valuable, but is often missing on many projects. The reason is that people start writing test cases before they have decided what they are going to test.

IEEE 829 - Test Case Specification

Test cases specify for each testing requirement:

- The exact input values that will be input and the values of any standing data that is required,
- The exact output values and changes of value of the internal system state that are expected,
- And any special steps for setting up the tests.

Defining the expected values is very important, for only by doing this can discrepancies be spotted. However in some projects they are not defined which results in a very poor quality set of test cases. A feature from the Test Design may be tested in more than one Test Case, and a Test Case may test more than one feature. The aim is for a set of test cases to test each feature from the Test Design at least once. Taking the Billing project example all three requirements could be tested using two test cases:

The first test case could test both that a normal bill is produced and that a volume discount is properly calculated.

A second test case could check that a final bill is produced and a volume discount is calculated.

Mandatory Part (Week 2)

For this lab, you can either install JUnit yourself or use the already installed JUnit on the CS server (recommended) - Check these tutorial

https://www.guru99.com/junit-tutorial.html

also check out the official site

https://junit.org/

In Java there are Math libraries containing various trigonometric functions. You will be tasked to write your own trigonometric library for Java. In doing so you will need to implement all the standard functions, SIN, COS, TAN. To calculate these functions consider the Taylor series expansions. You might find the following resource helpful:
https://www.efunda.com/math/taylor_series/taylor_series.cfm

It should be noted that these series are based in radian measure. It would be nice that your library could also generate the function values from degrees. Thus your library will require the appropriate conversion as well to allow the user to chain together functions to produce the desired result. You are not allowed to use any java math library functions when developing your

own. If you need a function, then you must create it. As part of your testing make sure that the functions are sound in all 4 quadrants.

As part of this assignment, you will be required to code the trigonometric functions sin, cos, tan in both radian and degree, integrating the Junit testing as you go. That is, you will also develop a test class under Junit to test the framework of your class as you implement the constructors and methods. When you are finished you should be able to run the test and verify that all cases have been covered and that your class is sound.

What this assignment is not: Turning in a working-class without any test suit will earn you a big fat zero. It is not about writing a java class, it is about learning Junit testing. It also means your class has to be correct. Thus, marks are assigned for completeness and thoroughness of the testing and the construction of the test environment.

Note : If you prefer programming in C++, you can use cppunit.

Methodology:
1. Write your test design specifications and test case specifications.
2. Write your trigonometry library.
3. Implement your tests, and run them.

Submit on blackboard your source code (class + test), test case specifications and test outputs/results (in a text file).

Optional Part (1% extra credit)

Read this page : https://www.guru99.com/continuous-testing.html
Look for Jenkins CI documentation.

Explain how you would integrate your testing suite in a continuous testing pipeline using Jenkins (illustrate your approach with screenshots).