ESOF3255 Software Testing & Quality Assurance
Project 1
Software Testing

1. Aims

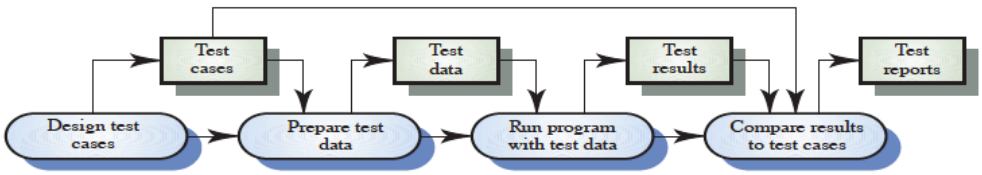
The aim of this project is to design and implement a testing plan using a number of complementary testing techniques. The aim is also to understand the difference between debugging and testing and to illustrate the problems related to software testing.

2. Objectives: On completion of this experiment the student should be able to:

1. use different testing techniques which are geared to find program defects
2. design *scenarios* that are representative of *test cases.*
3. show how test case design guidelines can be used for *exhaustive* testing.
4. explain the use of program structure analysis in testing

3. Background

One of the objectives of software testing is to uncover all program defects. A successful defect test is the one that causes a program to behave in an anomalous way. Tests show the presence not the absence of defects. The aim is to design *test scenarios* (based on *test data)* that are tailored to specific *test cases. Test data* are inputs which have been devised to test the system. *Test cases* however, are inputs to test the system and the predicted outputs from these inputs if the system operates according to its specification. The following figure highlights the process of test-case design for software testing.



Part 1 - Functional Testing

The *NextDate* program is a part of the *BlackBox.exe* (executable on a PC) described in the Readme.txt. All the necessary files can be downloaded from 'Project1.zip' available in the course website on D2L. The specification of NextDate is given below.

NextDate Specification

NextDate is a function of three variables: month, day and year. The output of the program is the month, day, and year of the next day. (Years are limited to those between 1812 and 2012 inclusive). Non-century years are leap years if they are divisible by 4; century years are leap years if they are divisible by 400.

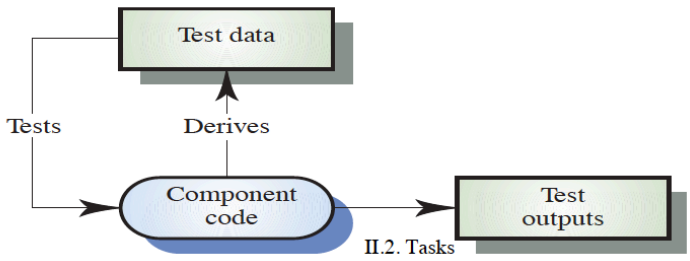
You are asked to complete the following tasks:

- Read and understand the *NextDate* specification.
- Decide what testing method, or methods, you are going to use. Provide a written explanation justifying your choice of the testing method(s).
- Prepare the Test Case Specification using the IEEE template as described in the IEEE Standard 829-1998 (http://ieeexplore ieee.oreie14/5976/16010/00741968.pdf). You may simplify the template by using only those specification items that may apply to this project.
- Using the Test Case Specification prepared above, test the *BlackBox.exe* and write-up the following Test Deliverables
    - Test Case Specification
    - Test Log
    - Test Incident Report
    - Test Summary Report
- Answer the following questions
    1. What confidence do you have that your test(s) have uncovered all the errors in the code? On what premises do you base your confidence level?
    2. Speculate what bugs you might have missed with your test(s) and why?
    3. Was the specification sufficiently detailed? Justify your answer.

Part 2

Structural Testing

Structural testing aims at deriving test scenarios according to program structure. The knowledge of the program is used to identify test cases. The Objective is to exercise all program statements, program paths, and combination of program paths. The figure below highlights this testing process.



For this part of the project you will have to design, implement and test the Binary Search algorithm. You may use either Java or C++ to build your *BinarySearch* Class and a test driver to perform the testing. To achieve this goal, you are asked to complete the following tasks.

a) Design and implement a *BinarySearch* class
b) Write its specification (clearly specify the preconditions and postconditions).
c) Test the developed *BinarySearch* class using the structural testing approach.
    - Decide what testing method, or methods, you are going to use. Provide a written explanation justifying your choice of the testing method(s).
    - Prepare the Test Case Specification (use the program flow graph to generate additional test cases)
    - Using the Test Case Specification prepared above, test the *BinarySearch* class and write-up the following Test Deliverables
        - Test Case Specification
        - Test Log
        - Test Incident Report
        - Test Summary Report
    - Answer the following questions
        - What confidence do you have that your test(s) have uncovered all the errors in the code? On what premises do you base your confidence level?
        - Speculate what bugs you might have missed with your test(s) and why?
        - Was the specification sufficiently detailed? Justify your answer.