University of Wollongong

SCSSE, Faculty of Informatics

CSCI222

System Development

Spring 2012

Assignment 1: "Testing"

Individual Assignment (10 marks)

Due 11:59pm, Friday 24th August 2012.

AIM

The aims of this assignment are to:

- Provide experience in the use of a modern Integrated Development Environment (specifically NetBeans running on a Linux platform) for the development of C++ applications;
- Provide additional experience in the design and implementation of simple C++ classes;
- Introduce the "Test Driven Development" style for program development;
- Use program testing tools including unit testing (cppUnit) and code coverage testing.

OBJECTIVES

On completion of this assignment, you should be able to:

1. Make effective use of an IDE for C++ development.
2. Develop a simple project in "test driven development" style.
3. Utilize the cppUnit unit test framework to run test suites on new components of application code;
4. Utilize the gcov code coverage tool to verify the extent to which a complete application has been exercised

The assignment does require that you use the NetBeans IDE as the standard C++ development environment. The Ubuntu Linux computer systems in laboratory 3.127 (and most of the other laboratories in building 3) will have this system installed.

TASKS

You should read the description of the DVD Selection System before reading these task details.

You are to implement the application described in those notes, and then produce a report on your application, and your development process. Your report should demonstrate that:

- appropriate use was made of the IDE,
- **you followed an iterative style of development creating and testing component classes prior to assembling them into complete programs**
- **you tested your final programs thoroughly.**

You prepare a report on your final code, your development process, and your testing. This report should be prepared in a word processor (Microsoft Word, or Open Office Word Processor) and should contain formatted listings of code, code coverage testing listings, screen shots, and segments of captured console output. The final report is submitted as a **PDF** document.

Use of the Integrated Development Environment

It is intended that you learn to use the IDE (and, if appropriate, learn to exploit the integrated source level debugger). Your report on your assignment work should include screen shots illustrating your code being developed in the IDE.

Classes and unit test exercises

You will need to implement a number of classes. In addition to application specific classes, you may need some standard classes from the STL and iostream libraries. Of course, there is no need for you to do any testing of such standard classes.

Remember that Test Driven Development (TDD) is more about design than testing. The TDD approach encourages you to think in terms of ***simple, orthogonal*** classes that can be developed and tested in isolation and which can then be assembled into a complete program.

You will create Test classes (these are all specialized CppUnit::TestFixture classes) that test the functionality of your application classes.

cppUnit classes, their more important methods, and the helper macros are covered briefly in the lecture materials. cppUnit is further documented at its SourceForge home[1].

Your assignment report should include formatted listings (header and code files) for your classes – both the application specific classes and their associated test classes[2]. If you do "refactor" your classes many times, you need only include final versions of the class and its test class. When presenting a class it is best to provide a brief summary of the role of the class, its header file, and then its implementation file.

You should do proper code-coverage checking of your program. You need to run your programs, possibly many times, to verify that you have executed all the different processing options.

---

[1] http://cppunit.sourceforge.net/doc/1.11.6/cppunit_cookbook.html

[2] Reminder: you can get formatted listing via the NetBeans "print as HTML" option.

Your report should include formatted listings of your application, some captured input and output from console sessions, and some gcov code coverage reports with your analysis/interpretation. You should edit segments out of your console sessions and code-coverage reports – *there is no need to include the full listings which can be very long*.

SUBMISSION

You are to submit a PDF formatted report file. This report file should be prepared in a word processor such as Microsoft Word or Open Office's word processor, and "printed" to PDF file[3].

The report is to have sections on each of your classes, and sections for the program(s) that make up the overall application.

Each "class" section should include the *complete* listing of the code of your class. In addition it should contain listings of the corresponding tests class. You should also include a report on your development strategy - e.g. were several iterations used to develop the class, was it necessary to revise the class at a later stage in the development project, etc.

The section on testing the program(s) should include some recordings showing interaction with the program (just the input commands and output responses as captured from a session), and edited excerpts from gcov profiling summaries **with your interpretation of the gcov results**. *The outputs from gcov are too lengthy to be included in their entirety, include only excerpts*.

BONUS[4]

The specification of the system in this assignment only covers a few basic functionalities of a real-life system. A maximum **1** *bonus* marks will be awarded if you can *identify*, *implement*, and *test* some other requirements of the system[5]. These should be *clearly* demonstrated in your report.

MARKING SCHEME

Before you get any marks for any component, there must be evidence supplied that the component works and has been thoroughly tested. Even if your component works, you may lose marks for coding defects.

- Overall presentation of report: **1 mark**
- Coding of applications and classes : **4 marks**
    - C++ usage (properly constructed class - header file definition, conditional include macros, appropriate style of member declaration, consistent naming conventions, consistent coding style, ...) : 0.5 mark

[3] Note that a well prepared report on this task might be a useful addition to the "work portfolio" that you are preparing to show potential employers.
[4] Note that the maximum marks awarded for this assignment are still 10.
[5] This exercise would give you some experience in eliciting requirements, a critical technique in the requirements phase and an important task for your future CSCI321 project and your future work in the industry. A bonus mark is only awarded to the implementation of **good and significant** requirements.

    - Classes: 2.5 marks
    - Overall application: 1 marks
- Effective use of testing tools and effective development strategy: **5 marks**
    - Unit test test suites for classes : 3 marks
    - Code coverage testing **(with your interpretation of the gcov results)** : 1 marks
    - Development strategy, use of iterative development style etc: 1 marks