COP 2535: Data Structures, Term Project

Step 04, Iteration 1, Testing

1 Introduction

The COP 2535 Term Project will give you the opportunity to build an application using one of the data structures we will learn about. You may in fact use several. This project is not intended to be a heavy-weight project, and I don't expect you to make an extraordinary on it. However, this could be part of your portfolio, so you could decide to make more of an effort if you choose to do so.

This project follows an iterative, incremental development model. It is *iterative* in that you will repeat the same series of steps four times through the term. It is *incremental* in that you will add to the functionality of your project during each of the four iterations. Each iteration will consist of the same four phases.

Requirements Analysis In this phase, you will determine the functional requirements to be implemented in this iteration. *Functional requirements* refers to what the software will actually do. This is probably the most important phase, because you cannot build anything unless you know what it is that you are building. Your deliverable will typically be one or more Use Cases, or a Functional Requirements Specification.

Design In this phase, you will design the software to be implemented. In some respects, this is the hardest phase. Thhis is where you will make decisions as to *how* to implement the requirements. Your deliverable will typically be in the form of UML diagrams. We will mostly be using Activty Diagrams, which are similar to flow charts.

Implementation In this phase, you will implement your design in code. The deliverable is your source code.

Testing In this phase, you will test your implementation against the requirements. That is, you will answer the question, "Does the code do what the requirements expect?" Your deliverable will typically be a text document containing the output of the code when run.

2 Instructions

Testing software is a critically important function of application development. This is not a software development class, and we will not cover testing. However, you will be required to demonstrate that your code compiles, runs, and meets your requirements.

Compiles and runs At the very minimum, your code should compile without errors, and should run to a successful conclusion. That is, it should contain no syntax ererors, no logic errors, and no run-time errors.

Meets requirements When you test software, you test against the requirements. That is, you test the code to see if it does what it is supposed to do. In a "real" development environment, you will work against a *Software Requirements Specification*, or perhaps a *Business Requirements Document*. These documents contain specific functional requirements for your application. This is not a "real" development environment, and we will not have an SRS or a BRD. However, you have Use Cases, and your software should follow the normal path.

Demonstration You will demonstrate your code in a live session. You should take between five and ten minutes to complete your demonstration. Your agenda will be as follows:

1. Briefly describe your Use Case, following the normal flow.
2. Briefly walk through your code, showing how it implements your Use Case.
3. Compile and run your code.
4. Discuss the output of your code or the results if no output.
5. State your intended Use Case for the next iteration.

3 Deliverable

Your delverable consists of a *plain text* file containing your source code *and* your output. That is, the file should have a `.txt` extension, nnd should be named something like `ProjectName_Testing01_Yourname.txt`. I expect one file, which includes your `main()` function. I *do not* expect anything fancy. Just very simple code that fully implements your requirements. Next week, you will start the second iteration with a new Use Case.