Task 2 Lab: C++ for Programmers

Summary:

You should all be programmers, but you may not be familiar or confident with C++ yet! If you are new to C/C++ programming, but you already have a strong programming language, work on finding the equivalent to what you already know. Even if you are confident, use this lab to quickly review your C++ skills so that you are ready for later work, and have good resources on hand to help yourself.

At the end of this lab you should have your IDE and workspace set up to compile, a working demonstration program for key C++ concepts, and experience debugging with C++.

Task Description

1. **Create a Simple Program in C++.** Create a simple program demonstrating the listed C/C++ concepts and features. Think of this as a tune-up check-list if you are quite familiar with C++ already. Store this in your repository in the appropriate folder.
    a. Create a function that accepts at least two values (simple parameter values, not user input), prints the values to screen, and returns nothing. Call this function from your main.
    b. Create a function that accepts one value, alters it, and returns it (simple value). Call this function from main, and print the returned value.
    c. Declare a pointer to a variable, and demonstrate the setting and reading of the value in the variable using the pointer.
    d. Create a function that uses a for loop to print all odd numbers between 0 and 20 on a single text line of text output, formatted with spaces. Call this from main.
    e. Create a function that creates a 1-dimensional array of 5 integers, then fill the array with random int values, print the contents of the array to the screen. Call from main.
    f. Split the line of text, such as “this has spaces in it”, by spaces, and show each part to screen.
    g. Declare a simple class that has at least one private variable, one public variable, and one public method. Create an instance of the class, and demonstrate its use
2. **Debug the provided Debug(gy) Application.** Effective use of the debugger is essential for isolating and repairing code errors within a non-trivial project of source code. Ideally you should be comfortable with using Visual Studio to compile and run a project. You should also be able to create a break point, step in/over/out, and inspect the value of variables while debugging a running program. In this lab you have been provided some basic code. Ideally you will learn from the code, but there are also some questions in the code that you must answer, and the debugging tools will help you to demonstrate your answers.
    a. Go through each #TODO question in the code (numbered), read the comments, follow the instructions.
    b. Change the Boolean “false” values to “true” for each “if” section.
    c. Uncomment particular lines if they are relevant to what you are trying to do.
    d. In your lab report, clearly state your answer to each question.
        i. There are some optional extra questions you might want to answer as well.
        ii. You can use screenshot images (suitably cropped) as evidence for key points, particularly when you are inspecting variables.
3. **Create a simple report.** Create a lab report document that will contain your notes about what you have done for this lab, and your answers to questions.
    a. Include your name, student id, the unit code, the task number and the date at the start of the report.
    b. We suggest using MS Word this time for easy image inclusion, but you could use markdown with images if you want a new challenge or prefer to do that.

Expected Output

Repository

1. Code (in repository) for Simple Program (#1)
2. Code (in repository) for Debug Application (#2)

Canvas

1. Report responding to the questions in the Debug Application (#3)
2. Commit logs

Notes

Commit to git as you go

As you progress, commit your work to your repository (see the git add, git commit and git push commands). Remember to keep your documents as well as your code in your repository.

You don't have to work alone

We encourage you to discuss your answers with other students in class and on Discord. Of course, your report and final code must have your unique work, but sharing ideas is a very effective way to consolidate what you learn.

Keep It Simple, Student

This lab is really just about making sure you are okay with inspecting variables with an IDE while also helping those with less experience with some more C++ code examples that will help with later work.