CS 240

Data Structures and Algorithms

Fall 2013

1 Lab 01

In this course you will be learning how to program in C++. Programming experience is a pre-requisite for this course, so in order for you to get your feet wet with C++, you are required to solve the following provided problems using C++. Please read the entire document and figure out exactly what I am asking for before proceeding.

2 Setup

The labs at Binghamton University, in `LN G103` are Ubuntu-based. There is also a Binghamton University Linux server that we can utilize in order to prevent you from having to physically be in the `LN G103` lab in order to do work. To connect to this server from your home machine you will need to use SSH.

2.1 Remote Access

For Windows users, you can get the SSH client from the University's FTP server, ftp://ftp.binghamton.edu/pub/windows/ssh2/, or you can use PuTTY, which you can get from the PuTTY Download Page, http://www.chiark.greenend.org.uk/ sgtatham/putty/download.html. For Mac and Linux users, there is an SSH client built in that can be accessed from a terminal session by typing `ssh username@hostname`. Linux users may have to install this utility manually.

The hostname of the server that we will be using is

`harvey.cc.binghamton.edu`

Your account credentials are the same username and password that you use to log into the `PODS` machines and any other Bingsuns account you have here on campus. If you are unable to connect to harvey, you may have to chain-SSH through `bingsuns.cc.binghamton.edu` in order to obtain an on-campus IP address initially.

IMPORTANT NOTE: You may find during the semester that you exceed your quota. This can happen if you are storing a lot of files on the Bingsuns server. If this is the case, I will not be able to help you, and it will take the computing center quite some time to resolve this issue so it is *your responsibility* to delete files that are not needed so that you remain below your quota. You can check how close you are to the quota by executing the `quota` command from a shell that is connected to `bingsuns.cc.binghamton.edu`. Exceeding the quota will make it so that you are unable to save your changes to files, compile your code, or sometimes it may even prevent you from logging into the machines during the lab session. Since you are expected to complete lab assignments during the lab sessions, failure to keep your data under the quota may impact your grade. Maintain your quota by checking it frequently and removing files from your H Drive whenever they are not needed. This includes clearing your cache in firefox and other browsers, as they will add hidden files to your H Drive.

You may also wish to use an FTP utility (either one of the SSH utilities listed above, or something like FileZilla) in order to transfer files between your local machine and the remote server. If you do not want to learn to use an FTP utility then you can learn to utilize a command-line editor such as nano, vim, or emacs. Each of these editors has some learning curve to them, but often the more steep the learning curve the more powerful the editor. During the lab sessions you may not have to use this editor (although it will still be available) and you can instead use a standard file editor like gedit. It is my recommendation to you that you use a command-line editor through SSH as it will produce the correct line endings (this is a problem when moving files from Windows to Linux and vice-versa), and it will also provide you with the experience to help you code more quickly during the lab sessions where time matters.

2.2 Using Linux

You can use the manual entry for most commands in Linux in order to figure out what they do, this is simply done by typing `man` and then the command you would like to know more about. Recall some of the Linux commands you should have used during Lab00 are:

- ls
- cd
- rm
- cp
- mv
- tar
- pwd
- touch
- nano
- vim
- etc.

3 Header Files

Once you begin the lab assignment, you will be required to implement some specific functionality. To make the programs modular, you will be given what is called a header file. Header files contain the declarations of functions, constants, variables, etc., and they usually have the name `filename.h`.

Header files often have what are referred to as Include Guards (http://en.wikipedia.org/wiki/Include_guard). These help to prevent multiple declarations and define scope. Header files are often laid out in such a way that the class is defined, including its functionality. This may look something like this:

```
class myClass{
     private:
          /*private data members and methods declared here*/
     public:
          /*public data members and methods declared here*/
};
```

Just like in Java, you may have to import functionality, whereas Java uses `import` statements, C++ uses `#include` statements. You may have to Google to figure out which file to include.

4 Implementation Files

Header files usually have a one-to-one correspondence between their associated implementation files, usually having the name of either `filename.cpp` or `filename.cc`. And most often the implementation files have include statements for their associated header files. For the `myClass` example, the header file would be named `myClass.h` and the implementation file would contain the line `#include "myClass.h"`. Implementation files are usually compiled separately and then linked together to produce a binary executable.

These files are where you will write your code. C++ has a scoping technique in order to differentiate between a function and a method. In general, if a class has a method, the method name will have to be scoped using the `::` scoping operator. Suppose you have a class `myClass` and it has a method `myMethod` with a void return type and an integer parameter, then your implementation file will look like:

```
void myClass::myMethod(int param){...}
```

If you are unfamiliar with the difference between a function and a method, this is an OOP concept that you should have learned, so you will need do a little self-study to determine the difference.

More information on this can be found in Appendix C of the Nyhoff book.

5 Makefile

For this lab and all future labs you will be required to write a makefile. You should have already completed the make tutorial found here,

http://mrbook.org/tutorials/make/.

It is a requirement for this lab that your makefile and your assignment utilize the following files:

- `Lab01.h` – Provided
- `Lab01.cpp` – Implementation file that implements all of the functions in the associated header file.
- `Driver.cpp` – Contains the entry point to your program, a sample file with this name will be provided. You are to use the main method to write test cases for your implementation and check that it does what it needs to do. *This testing should be done before lab as you do not want to waste lab time trying to fix errors that you should have fixed while working on the pre-lab.*

6 The Questions

For each question, before you look at the associated header file that you are to implement, you should consider what your method signature will look like. This will help you to solidify what each question is asking before you begin answering.

Question 1

Find the largest integer value in an integer array. In this question, we will pass the integer array (`int []`) and an integer capacity for the array. You are to return the integer that has the maximum value amongst all values in the array. In the event that you cannot find the maximum value, you should return `-10000`.

Question 2

Write a method that prints out only the even numbers in the range from A to B (exclusive), one per line. Note that this means A and B must be parameters to your method.

Question 3

Write code to find the smallest common multiple of two positive integers. The parameters for this method should be the two integers. In the event that no such multiple exists, you should return `-1`.

Question 4

Write a method to reverse a string. You should use C++ style strings in order to perform this task. You can find more information on these type of strings at http://www.cplusplus.com/reference/string/string/?kw=string.

Question 5

Write a method that prints the numbers from 1 to 100, one per line. For multiples of four, instead of the number, print the word `Fizz`. Similarly, for multiples of seven, print the word `Buzz`. For numbers that are multiples of both four and seven, print the word `FizzBuzz`. Note that capitalization and spaces matter; reproduce these words exactly.

Question 6

Write a method that will print the grade-school multiplication table up to 12. Mark the upper left-hand entry of the table with an `X` and separate the entries with a tab character. It will be easier to program this if you utilize multi-dimensional arrays. Refer to section 3.3 in the textbook to understand more about multi-dimensional arrays in C++.