CSE5/7343 - Programming Assignment 1

Due Sunday, February 5, 2017 at 11:59 p.m.

Uploaded to Canvas

Projects in this class will be implemented in the C language. Java borrows heavily from C syntax, and it is relatively easy to learn one language if you know the other well.

This programming assignment is to help you get up to speed on the language before we start using it for interesting things in the next programming assignment. It has two parts. First, do some reading of a more extensive (and better!) discussion of the differences between the languages. Second, write a few **simple** programs to begin practicing a few of the mechanics.

1. Reading

Here are some other useful resources. You can find more with some simple web searches.

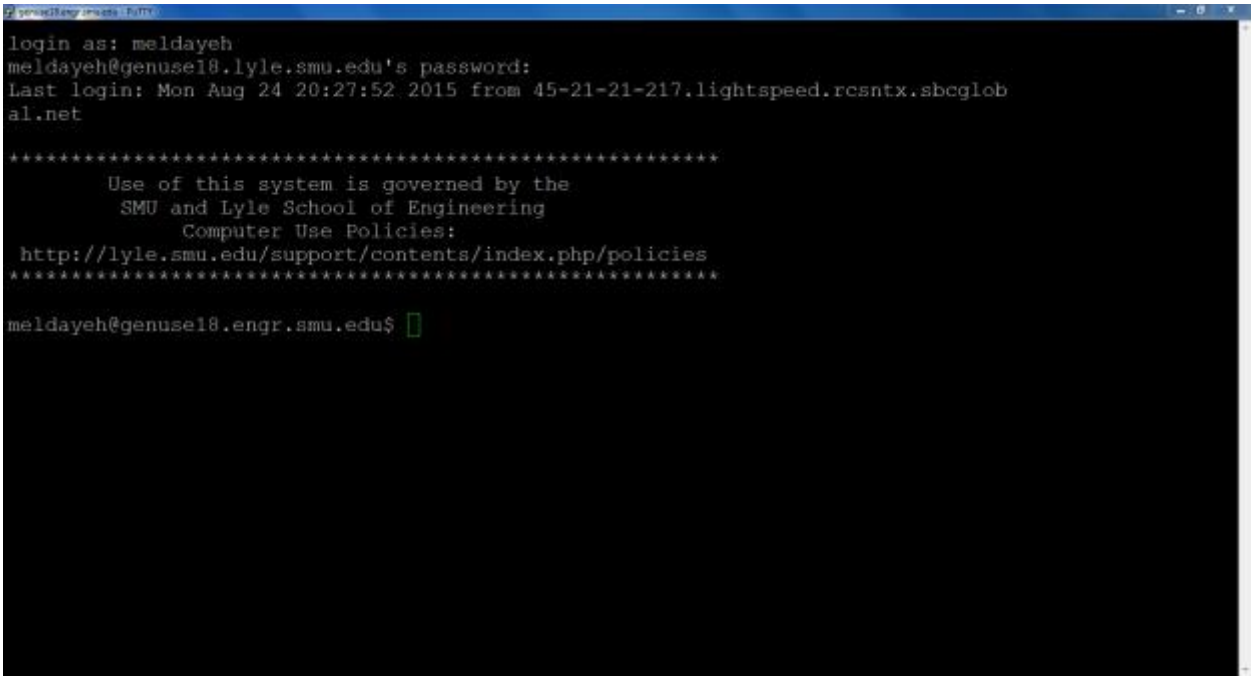
- C for Java Programmers (Cornell)
- Brian W. Kernighan -- Programming in C A Tutorial

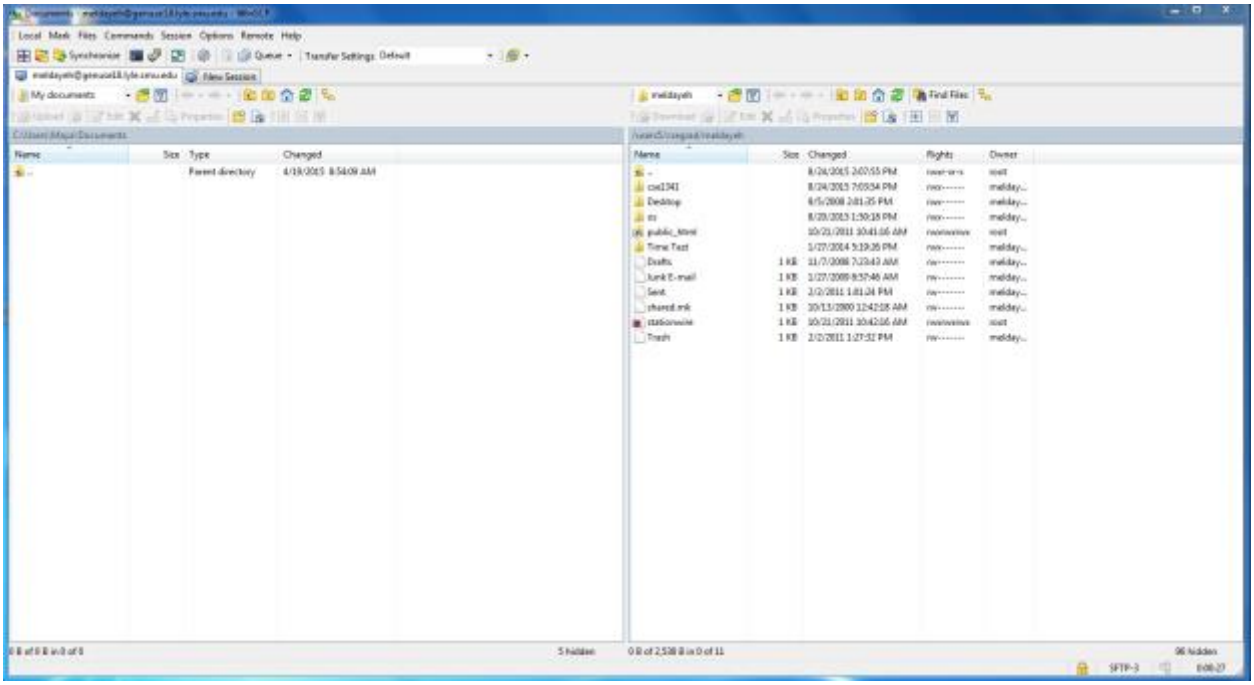
2. Programming

To help you get familiar with C, in this programming assignment you will write some simple C programs.

2.1 Setup

You will implement this programming assignment and upcoming programming assignments on the General Use Linux machines provided by the Lyle School of Engineering. If you do not have an account yet, click here for instructions on how to request an account. To login remotely to the school's Linux machines from your Windows machine, download PuTTY and WinSCP (installation package). PuTTY will allow you to compile, link, and execute your code remotely and WinSCP will allow you to drag and drop files from/to your machine and the Linux machine.





Although most or all this programming assignment should "just work" in many other environments (cygwin, OSX, solaris, etc.), note that (1) I will not be able to assist in setting up or debugging problems caused by differences in the environment and (2) statements like "it worked on my home machine" will not be considered in the grading process. If you choose to do development in an unsupported environment, it is *your responsibility* to leave adequate time to port your solution to the supported

environment, test it there, and fix any problems that manifest. You will use Canvas to upload your files and submit your assignment.

You should be able to compile and run the test program hi.c which is found in the folder I provided with this programming assignment.

Login to your account on one of the Linux machines using PuTTY. At the prompt, which should look like <yourusername>@genuse<machine#>.engr.smu.edu$,type the following:

```
$ mkdir cse7343
$ cd cse7343
```

Download the zipped folder ProgrammingAssignment1. Unzip it and use WinSCP to drop the folder inside the directory you just created. Then, perform the following steps:

```
$ cd ProgrammingAssignment1
$ gcc -o hi hi.c
$ ./hi So far so good.
$
```

(0) Name

Before you start writing code, create a text file <yourname_ID>.txt in the programming directory and put your full name and ID number in there. Include this file when you turn in your programming assignment. Do not turn in your assignment as separate files. Move your assignment to your local machine and zip the folder. Attach the zipped folder to your assignment submission on Canvas.

(1) Hello

Write program hello.c that prints out the string "Hello world\n".

```
$ gcc -o hello hello.c
$ ./hello
Hello world
```

(2) Loops

Write a program called *words* that prints out the words from the command line on different lines

```
$ gcc -o words words.c
$ ./words To be or not to be? That is the question.
To
be
or
not
to
be?
That
is
the
question.
```

(3) Procedure calls

Write a program fact.c that uses recursion to calculate and print the factorial of the positive integer value passed in or prints the line "Huh?" if no argument is passed or if the first argument passed is not a positive integer. If the value passed in exceeds 12, you can simply print "Overflow".

```
$ gcc -o fact fact.c
$ ./fact one
Huh?
$ ./fact 5
120
$ ./fact 5.1
Huh?
```

(4) Headers, Linking, Structs

You can split your code across multiple source files. Typically, a header file (e.g., "foo.h") describes the procedures and variables exported by a source file (e.g., "foo.c"). To compile, each .c file is typically compiled into an object file (e.g., "foo.o" and "bar.o") and then all object files are linked together into one executable.

You can adopt a basic object oriented style of programming in C (even without the syntactic sugar of C++) by defining a type and the methods that operate on it in a .h file and a corresponding .c file.

I have provided point.h, which defines a type and structure for storing a point's position in 2d space, and which defines the interface to a *translate()* function to move the point

to a new location and to determine the *distance()* between points. Your job is to implement these functions in point.c so that the test program testPoint.c works. For a tutorial on pointers in C, click here.