COMP 2404 -- Tutorial #1

Linux and Basic C++

Learning Outcomes

After this tutorial, you will be able to:

- understand a basic C++ program
- compile a program using a Makefile
- use redirection of standard I/O in Linux

Instructions

1. If you are using your own laptop, make sure that you have installed VirtualBox and the extension pack. You must download the course VM from the SCS web site, and you must import the VM into VirtualBox. The instructions for this process can be found here.

2. Start up the VM for the course and log in. Try some of the basic commands from the course to find your way around the Linux file system. Sample commands to try out:

`pwd`	present working directory
`ls`	list directory
`ll`	list directory with options -alF
`cd`	change directory
`mkdir`	make directory
`cp`	copy file

   Try using wildcards with your shell commands. Use the manual (`man`) pages to get help on any of these commands. Example: `man ls`

3. Create a COMP2404 directory in your home directory, and a Tutorials sub-directory:

   ```
   cd
   mkdir COMP2404
   cd COMP2404
   mkdir Tutorials
   cd Tutorials
   ```

4. From the tutorial page in *cuLearn*, download the tutorial archive file `T01.tar` into your Tutorials directory.

   Un-tar the file, using the extract, verbose, from file options:

   ```
   tar -xvf T01.tar
   ```

   List the content of the directory, and you should see all the tutorial files.

5. Using the text editor of your choice, or the `more` or `less` commands, look at the content of each file, beginning with `main.cc` and the `Book` class files. You won't be able to understand every line at this point, and that's normal. You can still get a good understanding of what the program and each function does. You will be working with this program and building on it for all 10 tutorials in this course, so it's important that you understand how it works.

6. Compile the program using the given Makefile. Run the program several times with different input to get an understanding of the expected input and output. Note that if you input a string where integers are expected, the program will get stuck in a loop. Try it out for fun! You can always kill a program using <Ctrl>-C. Converting strings to numbers is a problem that requires the `stringstream` class, which we will learn about in a later tutorial.

7. Look at the content of the `in.txt` file. How does it compare to the input that you've been typing in? Using pipelining, connect the program executable's standard input to `in.txt`:

   ```
   t01 < in.txt
   ```

   Save the standard output of the program executable by connecting it to a file called `out.txt`:

   ```
   t01 < in.txt > out.txt
   ```

   Look at the content of the `out.txt` file to make sure it's what you expected.

8. Copy the existing `in.txt` file to a new `in2.txt` file. Using a text editor, modify the new `in2.txt` file so that at least five more books are added. Run the program again with the new `in2.txt` file, and check that the new books are printed out at the end of the program.

9. Using the Makefile's `clean` target, clean up the directory to ensure that you don't save any unnecessary files.

10. Package together the tutorial code and your new `in2.txt` file into a new tar file, using the create, verbose, to file options:

    ```
    tar -cvf myT01.tar *
    ```

11. Start up a browser in the VM, log into *cuLearn*, and go to the tutorial page. Select the tutorial 1 submission link, and upload your new tar file.

12. Save your work to a permanent location, like a memory stick or your Z-drive.