Lab 1 - Introduction to Linux and C++

Due: end of class

Review

Recall that the Linux file system is tree-structured, with files and directories.

Some useful linux/unix CLI (command line interface) commands are:

- man <command>: description of command, including all its options
- ls: list files in current directory.
- pwd: print working directory (i.e., current directory)
- cd <dir>: change directory. Some useful values of <dir> include . (current), .. (parent), / (root of file system), and ~ (home)
- mkdir <dir>: make a directory in your current working directory

If you need a refresher, there is a simple tutorial here.

Also recall that some standard editors for Linux include gedit, emacs, and vim. You may use any of these, though emacs and vim are most common in the CS world.

Task 0

To avoid cluttering your home directory, create a directory titled lab1. The instructor's home directory contains a small program whose name starts with "." and ends with "1a.cpp". If you don't know how to find the file, use man to find the proper ls option (and/or use wildcards). Find this file, copy it to ~/lab1/lab1a.cpp, compile (using g++ -o lab1a lab1a.cpp), and run it (./lab1a). What does it output?

One problem with the program is that it does not output a carriage return (so, the prompt for the next command is on the same line). Edit the program to add your instructor's name at the end of the output, and also fix the above problem. Rerun.

Task 1

Write a program in ~/lab1/lab1b.cpp that outputs "Good morning <instructor>.", "Good afternoon <instructor>.", or "Good evening <instructor>." depending on whether the current time is midnight-noon, noon-sunset, or sunset-midnight respectively.

Your program should first prompt the user for the sunset time with the following prompts, inputting data using cin (e.g., "cin >> cur_hour") after each prompt:

```
Enter the hours part of today's sunset time (4-9):
Enter the minutes part of today's sunset time (0-59):
```

For this assignment, you may assume that the user enters a legal value, and you should not do any error checking (since this is just the first day of class!). These values obviously won't work in the Arctic, but we won't worry about that either.

You will also need to retrieve the current time (24 hour format). A standard platform-independent way to do this is through the ctime library. Since you have not covered enough C/C++ to use this yet, you can just cut/paste the following sequence to get the current hour/min:

```
time_t t;
```

```
   struct tm *now;
   t = time(0);              // get current
time
   now = localtime(&t);      // adjust for local timezone
   int hour = now->tm_hour; // retrieve current hour
   int min = now->tm_min;   // retrieve current min
```

You will also need to include the ctime library (#include <ctime> at the top of your file).

In your submitted program, you should also state the test cases you tried, and why you picked those. Write these in *brief* comments.

Task 2

Modify the above program as follow:

- Add a prompt asking "How many minutes from now do you expect to be home?", and output a sentence saying "You will get home at HH:MM".

SUBMISSION

Your instructor will give you directions on how to submit the program.

You should have the following header on all programs this semester:

```
/*
  Author: <name>
  Course: {135,136}
  Instructor: <name>
  Assignment: <title, e.g., "Lab 1">

  This program does ...
*/
```

GRADING

All 135 and 136 programs this semester will be graded on:

- Correctness: Does your program work?
- Testing: Have you generated sufficient and good test data to give reasonable confidence that your program works?
- Structure: Have you structured your code to follow proper software engineering guidelines? This includes readability and maintainability. Note that
- Documentation: How well documented is your code? Good documentation does not repeat the code in English, but explains the point of each code block, highlighting any design decisions and/or tricky implementation details.