Mechanics GCC

CSCI-243 Homework

5/14/2018

1 Introduction

In this assignment you will practice using the CS department's Ubuntu machines. All homework and projects will be required to compile and run on these machines.

You will need to use a terminal window and UNIX/ Linux programming tools to get materials, edit code and text files. In addition, you will need to use the CS department's programming tools, get, and eventually gmakemake to do your assignments.

2 Resources

Here a a list of common resources you will need throughout this course.

2.1 Unix/Linux

You need to become familiar with the basics of Linux (and, by extension, other UNIX-like systems) to do your assignments this term. Unless you are already very experienced with Linux, you should complete at least parts 1 and 2 of the http://www.ee.surrey.ac.uk/Teaching/Unix/ tutorial.

2.2 Editors

On our Ubuntu Linux systems, there are several editors available, including:

- pico
- nano
- gedit
- nedit
- vi
- vim

The `gedit` and `nedit` are graphical, while the others work inside a terminal window.

We recommend learning `vi` or `vim` due to their extensive use in the UNIX/Linux community. The `emacs` editor is another powerful, versatile editor, although fewer faculty and students in the department are experts.

The http://www.openvim.com/tutorial.html covers the basics of vi and vim. You also can take the http://www.openvim.com/tutorial.html tutorial that teaches the editor using an adventure game.

2.3 SSH: Secure Shell for Remote Access

When you cannot come to the CS labs to work, you need to use a secure shell (SSH) client to connect from a remote system such as your own computer to a CS department Ubuntu machine with the correct configuration.

You need to do a remote login from your machine and use your CS account, not your institute DCE account to connect. When logged in through ssh, you will have to use a plain text editor that has no graphical interface unless you learned to run the X11 Window System on your local computer and enabled X11 within putty/ssh.

Refer to the CS howtos page for assistance.

2.4 Windows Linux Integration

Windows has an integrated Linux shell. This will simulate an Linux environment on your Windows machine. To install this feature you must have Windows 10. Instructions for this are located at https://docs.microsoft.com/en-us/windows/wsl/install-win10.

You will need to install GCC. Start the Linux shell and type `gcc` and it will give instructions to install it.

3 Account Setup

It is time to create a useful directory structure for storing your assignments. You should not use the "WIMP" (window, icon, mouse, pointer) techniques with which you may be most familiar. Instead, you should navigate, create folders, list them and determine your working directory using these shell commands inside a terminal window:

- cd
- mkdir
- ls
- pwd

Create the following directory folder structure under the Courses subdirectory of your CS account's HOME folder. Plain text and indentation below indicates directory hierarchy. This should be the output of the tree program after you cd Courses and enter the command shown here:

```
$ tree ./Courses/CS243
./Courses/CS243
|-- Homeworks
|   |-- 1
|   |   `-- your-hw1-files-go-here
|   `-- 2
|       `-- your-hw2-files-go-here
`-- Projects
```

```
    `-- 1
        `-- your-project1-files-go-here

5 directories, 3 files
```

For each subsequent assignment, you will `mkdir` the appropriate folder in the correct place and use that as the location for your work on that assignment.

4 Assignment Tasks

This section describes multiple activities to complete for credit on this assignment. Always do the activities in the order in which they are listed.

4.1 Studying Erroneous Code (30 points)

For this activity you need:

- questions1.txt: A file containing questions to answer; and
- warning.c: A complete C program to fix.

These items can be obtained from the Homework section in myCourses for Homework 1.

The standard command and command line arguments you should normally use to compile your code in this course are:

```
gcc -std=c99 -Wall -Wextra -pedantic
```

The names of the C source file(s) you wish to compile must appear at the end of this command.

If you are compiling to create a .o file and not compiling to create an executable file, you must add this option flag to the list:

```
gcc -std=c99 -Wall -Wextra -pedantic -c
```

If you are compiling to create an executable file, you should use this flag and filename sequence:

```
gcc -std=c99 -Wall -Wextra -pedantic -o <filename>
```

That will specify the file name that will contain the executable. (Future lectures will cover these options in more detail.)

After you figure out the problems with the program, fix them, and edit the questions1.txt file to add your answers to the questions.

You will submit your `warning.c` and `questions1.txt` to myCourses as part of your solution.

4.2 Linking Compilation Units, Header Files (30 points)

In this activity you need:

- questions2.txt: A file containing more questions to answer;
- main.c, circle.h, circle.c: A complete program that has compilation problems to diagnose, explain and fix.

These items can be obtained from the Homework section in myCourses for Homework 1.

Read the code carefully. Try to compile it. After you have figured out the problems with the program, fix them, and edit the questions2.txt file by entering the answers to the questions.

You will submit the your `circle.h`, `circle.c`, `main.c`, and `questions2.txt` to myCourses as part of your solution.

4.3 Implementing a Simple Algorithm in C (40 points)

In this activity you need to complete:

- triangle.c: The skeleton of a program that prints text-based triangles of various sizes.

This item can be obtained from the Homework section in myCourses for Homework 1.

Complete the function that prints right angle triangles. The triangles should be filled with asterisks, and the program output should look like the following:

```
*

    *
  ***
*****

      *
    ***
  *****
*******
```

If the function's argument is even, the function increments the size by 1. The main program should print the 3 triangles one right after and the other.

Your program must use loops; you may not hardcode the statements to print each of the specified triangles, and you may not use recursion.

You will submit your `triangle.c` to myCourses as part of your solution.

5 Submission Instructions

Put a zip file, `hw1.zip`, containing the files for this assignment into the correct MyCourses drop box.

6 Grading

- 30%: Studying Erroneous Code
- 30%: Linking Compilation Units, Header Files
- 40%: Implementing a Simple Algorithms in C