Introduction to Linux and C++

Topics: Terminal Commands, compiling with g++, C++ basics
Lab 1: https://maryash.github.io/135/labs/lab_01.html

- Sadab Hafiz

Things you are expected to know already:

This course is assuming that you have some prior knowledge about programming. Most of the students are coming from CSCI 127 which covered basics in C++:

- C++ basic syntax (variables, libraries, comments)
- Input and output in C++
- Conditionals in C++ (if-statements)
- C++ for-loops
- Basic math operations
- Common Unix/Linux terminal commands

If you know these concepts in a language other than C++, try them out in C++ and get familiar with C++ syntax. Everything in this course will be done using C++.

C++ basic syntax (recap)

The following is a very basic C++ program that prints “Hello World!”:

```
#include <iostream>                 // import iostream library (since we are using `cout`)
using namespace std;                // use standard namespace to avoid typing “std::”
int main() {                        // main function definition
     cout << "Hello World!";        // print “Hello World!” using cin
     return 0;                      // end of
}
```

Detailed Explanation: https://www.w3schools.com/cpp/cpp_syntax.asp

C++ Variables (recap)

Variables in C++ are assigned a datatype during initialization. Most common datatypes are: int, string, float, double, long etc. Usage:

```
#include <iostream>

using namespace std;

int main() {

     int num = 2;                                // integer variable named `num`

     string greeting = "Hello World!";           // string variable named `greeting`

     return 0;

}
```

Datatypes: https://www.tutorialspoint.com/cplusplus/cpp_data_types.htm

C++ comments (recap)

Programmers explain and document their code using comments. Single-line comments start with two forward slash: `// This is a single line comment`

Multi-line comments start with “/*” and end with “*/”. Follow this template in the beginning of all code that you submit for this course:

```
/*
Author: your name
Course: CSCI-136
Instructor: their name
Assignment: title, e.g., Lab1A


Here, briefly, at least in one or a few sentences describe what the program does.
*/
```

Input and Output in C++ (recap)

Basic input and output in C++ are part of the <iostream> library. Get inputs using `cin` and `cout` is used for output. For example:

```
#include <iostream>                                              // required library for `cin` and `cout`

using namespace std;

int main() {

      int num;                                             // integer variable named `num`

      cin >> num;                                          // get the value of num from user input, e.g. 5

      cout << “You entered the number ” << num << endl;      // outputs “You entered the number 5”

                                                           // `endl` is used to go to the next line

      return 0;

}
```

if-statements in C++ (recap)

You should be familiar with the usage of if-statements or conditionals. The order of conditions matter. Here’s an example of if-statements in C++:

```
if (num == 1){                                                      // if num is equal to 1

      cout<< “You entered 1” <<endl;

}

else if (num == 2){                                                 // if num is equal to 2

      cout << “You entered 2” <<endl;

}

else{                                                               // if num is not 1 and not 2

      cout << “You entered a number that is not 1 or 2” << endl;

}
```

for-loops in C++ (recap)

If you want to repeat a task over and over again, use loops. The most common loop is the for-loop. Here’s an example:

```
// i is an integer variable that changes in each iteration of the loop

// i (start = 0; stop = 3; step = 1)

// i will be 0, 1, 2    (notice it doesn’t reach 3, since it starts from 0, repeats 3 times)

for (int i=0; i<3; i++) {                                   // repeat 3 times

     cout << “The value of i is ” << i << endl;             // outputs value of i

}

/*   Output:    The value of i is 0

                The value of i is 1

                The value of i is 2                                            */
```

Basic math operations (recap)

C++ supports addition subtraction, division, multiplication and many more! For now, you should already know how to do the basics:

```
# include <iostream>

using namespace std;

int main() {

      int a = 2;                                // a = 2

      int b = a + 3;                            // b = 2 + 3 = 5

      int c = (b + a) * 10;                     // c = (5 + 2) * 10 = 70

      int d = c / 5;                            // d = 70/5 = 14

      return 0;

}
```

The modulo operator (%)

The modulo operator (%) computes the remainder of a division operation. For example, 37 % 10 returns 7, because this is the remainder of 37 when divided by 10. This can be very useful for different tasks. Here’s an example:

```
int num;

cin << num;                                    // get input from user

if (num % 2 == 0){                             // if num divided by 2 has remainder of 0

     cout << “Even number” << endl;            // it is an even number

}

else {

     cout << “Odd number” << endl;             // otherwise it is an odd number

}
```

Basic Unix/Linux commands (recap)

Command	Description
pwd	print the current working directory
ls	list files in the current directory
ls path/to/a/directory	list files in the directory
cd path/to/a/directory	change directory
cd ..	go to the parent directory (one level up)
mkdir newdirectoryname	create new directory
cp file1 file2	copy file1 and call the copy file2
mv file1 file2	rename (move) file1 to file2
rmdir directoryname	remove empty directory
rm file	remove file

Additional Unix/Linux commands and shortcuts

The following directory shortcuts can be combined with other commands:

Shortcut	Description
.	the current directory
..	the parent directory of the current
~	the home directory

Additional terminal commands:

Command	Description
touch newfilename	create new file
chmod <options> file	change file permissions (read +r, write +w, execute +x)
man command	documentation about the command

More information: http://www.ee.surrey.ac.uk/Teaching/Unix/unix1.html

Compiling code with g++ in the terminal

In Linux/Unix terminals, `g++` allows compilation of C++ code. Let's say that we have some code written in a file called `code.cpp`. If we compile this using `g++`, `g++` will create an executable file.

We can configure the name of this file during compilation:

- Compile with default executable name: `g++ code.cpp`
- Compile with custom executable name: `g++ -o myprog code.cpp`

Running the executable:

- Default executable: `./a.out`
- Custom executable: `./myprog`

If you're on windows, there are different ways to get g++ working on your machine.

Try: Linux on Windows Tutorial or C++ programming with Visual Studio Code

Which text editor to use?

There are different IDE and text editors out there that can be used with C++. Most common ones are: VS code, Vim, eclipse, sublime text, gedit etc.

Try out some of them and see which one works for you. I personally recommend using VS code since it works with different programming languages and supports different plugins.

Check the course webpage daily for updates and materials. Each homework and lab prepares you for the quizzes and exams. Keep up with the deadlines and due dates for all assignments and you will be prepared for the exams and quizzes.

Course Webpage: https://maryash.github.io/135/index.html

Good Coding Practices

Computer Science is a very competitive field. What separates a good programmer from the competition is how readable their code is. This includes being consistent in syntax, descriptive variable names, proper documentation and comments etc.

While your code might compile and run, it maybe difficult to understand for other programmers. Being consistent will help both you and the UTA that is trying to help you debug your code. Try to follow this style-guide:
https://maryash.github.io/135/worked_examples/style_guide.html

Go to recitation sections with questions regarding the material and ask UTAs for help if necessary.