CSCI 205 Programming in C/C++
Fall 2020

Class meets: MWF 2:00 – 2:50 p.m. Zoom Discussion and Recitation Sessions

Professor: Dr. Melissa Holmes
email: melissa.holmes@umontana.edu
office: Social Sciences 411
text messages: 406.565.6079
office hours: Listed under "Getting Help" on Moodle

Course Description:
The C language is a very powerful, flexible and well-established language. It was originally developed as a high-level (human-readable) language that is low-level enough to do things such as write operating systems, control hardware, etc. Many derivatives of the language have been developed, such as C++, C#, etc. C++ is a version of C that supported objects and classes. This class will cover both C and C++.

Grading:

Short programming assignments, quizzes and problems	60%
Exams (2)	20%
Large programming project	20%

Textbook:
None required. A variety of online resources will be used.

Accommodations:
Students who need any type of accommodation should work with Student Disability Services and provide appropriate documentation as soon as possible.

Academic Dishonesty:
You are encouraged to work in teams and use many resources including books and the Internet. However, each student must turn in his/her own work, and each student is responsible for understanding anything that is turned in. Refer to the Student Conduct Code for more information regarding plagiarism and cheating.

Student Learning Outcomes: Upon the successful completion of this class, students will be able to:

1. read a problem specification and define functional requirements for the problem;
2. articulate the purpose and paradigms of the C and C++ languages
3. write C and C++ programs of intermediate complexity that implement common data structures
4. run a C program on a Raspberry PI
5. use pointers and dereferencing

Course Schedule (Subject to Change)

Week	Topics
0 Aug. 19- 21	Syllabus, IDE and GCC, Intro to C – history, purpose, paradigm, syntax, first programming assignment
1 Aug. 24 - 28	Assignment, decision, repetition, functions, data types
2 Aug. 31 – Sept. 4	Strings, enum, struct, compilation
3 Sept. 7 - 11	Arrays – declaration, traversals, array of struct More functions – pass by value/method
4 Sept. 14 - 18	Pointers and dereferencing, dynamic allocation of memory
5 Sept. 21 - 25	Linked list in C, traversals using pointers Exam Review
6 Sept. 28 – Oct. 2	Exam 1 Other data structures in C (stack, queue, etc., traversals)
7 Oct. 5 - 9	C++ general overview, intro objects and classes
8 Oct. 12 - 16	OOP in C++
9 Oct. 19 - 23	OOP programming project
10 Oct. 26 – 30	Begin Raspberry PI projects
11 Nov. 2 - 6	Raspberry PI projects
12 Nov. 9 - 13	Raspberry PI project
13 Nov. 16 - 20	Final project of your choosing (large programming project that optionally uses Raspberry PI)
14 Nov. 23-25	Appointments to present final projects; final exam paper due.

CSCI 205 Coding Standard

Comments

There should be 3-5 lines of comments at the top of each file. This heading should include your name, the class and semester, the lab # and title and the date. If it is part of a package you can include more information as needed.

Example:

```
/*
        Melissa Holmes
        CSCI 205 Fall 2020
        Lab #1 Fortune Picker Lab          September 4, 2020
*/
```

Short comments throughout the code should describe the tasks of the program, i.e. "get n1 from the user." The purpose of some variables should be described, if not clear.

Do not clutter your code with unnecessary comments.

Whitespace

A line of whitespace should exist between the tasks of the program.

Indenting should be used to indicate nesting in blocks of code. Spaces, not tabs, should be used to indent. Modern IDE's have a setting to save whitespace as spaces.

Braces

When curly braces are used to indicate blocks of code, the first brace should appear on the same line as the code, and the closing brace should appear on its own line, i.e.:

```
if (condition)   {
        //lines of code

}  //closing brace on its own line
```

If appropriate for clarity, closing braces should be labeled with comments.

Identifier Names

Use identifier names that are meaningful, i.e. "firstSelection" may have more meaning than "a" or "n1"

Camel case should be used for variable and method names, i.e. theFirstVariable, myMethod

Camel case with the first letter capitalized should be used for class names, i.e. HelloWorld