Last updated 27 September, 2019

CS133C – Programming in C
Fall 2020, CRN: 26949, Credits: 4

Instructor: Wendy Roberts

email: robertw@linnbenton.edu

Initial class note: This is **not** an introduction course to computer science (CS), though that does not mean it will be difficult. This is an intro to the C language for those who have some programming exposure and an expansion on the design, testing, and implementation concepts covered in an introductory CS course using programming as a tool. Do not expect a gentle intro to programming, design, testing, C, or other tools that we use; instead, expect to do some reading, much practicing, and much discussing of the topics we cover (though we will have a lab to make this considerably easier on any time taken away from the rest of your life).

1. LBCC catalog course description, including pre-requisites/co-requisites:

Introduces problem analysis and programming to solve computation problems. Introduces the C language for those with previous programming experience.

Prerequisites: CS161 Intro to Computer Science I with a grade of "C" or better or equivalent experience as determined by a Computer Systems Department instructor; MTH 095 Intermediate Algebra with a grade of "C" or better

2. Class Time-space:

2.1 Lecture + demo:	MW	1200 – 1320,	Online
2.2 Lab:	F	1200 – 1350,	Online

3. Course Objectives:

Upon successful completion of this course, students will be able to:

3.1 Demonstrate the use of good program development techniques and documentation in the C programming language.
3.2 Demonstrate an understanding of data types and their representation in the C programming language.
3.3 Write and debug C code that includes control statements, while loops and for loops.
3.4 Write and document C code that includes the use of one-dimensional arrays.
3.5 Write, debug and document C code that makes use of pointers and linked lists.

4. Learning resources:

Note: Texts are not required but recommended.

4.1 **C Programming Language (classic resource for C Programming)** - *by Kernighan and Ritchie*

ISBN 978-0-13-110362-7
Copyright 1988
Publisher Prentice Hall
Edition 2

4.2 **The C Book – Open Source GBDirect Publications**

4.2.1 **https://publications.gbdirect.co.uk//c_book/**

4.3 **Accessible storage** Minimum capacity 16MB.

USB flash memory is easily available,
Cloud and other Internet storage should be accessible.

4.4 **The C Language –** I strongly recommend using portable code. We'll discuss this more in class.

4.5 **A C Compiler -** I prefer Visual Studio, but you are welcome to use the compiler of your choice (such as gcc, visual studio, code::blocks, or Xcode). There is also a good online version at https://www.onlinegdb.com/online_c_compiler. The goal is for you to get a low-level understanding of our code assignments by carefully writing our implementation.

4.6 **A student Github account, shared with the Instructor.**

4.7 A strong desire to learn and experiment with the C Programming language and software design principles.

5. Grading:

5.1 Scores for coursework items will be available upon grading completion.

5.1.1 My primary grading in programming classes is to have students show me where they are throughout the week so we can discuss approaches and any requirements that would affect the grade.

5.2 Students will be required to turn in all coursework items before **23:59 (Pacific Time Zone)** on the date that they are due (**generally Monday in my courses**).

5.2.1 Students must be sure to give themselves plenty of time to submit coursework, as late work will not be accepted without prior consent or special circumstances. Please do not start your coding on the day the assignment is due. Coding a working program almost always takes longer than anticipated.

5.3 To earn a passing grade in this course you must pass each of the following coursework categories:

5.3.1 **Demonstration**: Discussion and weekly assignments – 50%

5.3.1.1 There are several projects to be completed for this class that are designed to challenge and solidify design, coding, and testing skills.

5.3.1.2 Project components are generally graded based on:

- completeness (does it compile and run)
- correctness (does it meet the listed requirements)
- quality and explanation of the design (features in advance, organized)
- quality and explanation of the tests (test each feature and expected successes and failures)
- quality of the implementation (consistent and readable style, runs well, is easy to learn and use)
- Note: careful design, systematic testing, consistent style, and readability of code are important software quality factors (all of which are subject to interpretation but graded by the instructor based on the spirit and letter of the requirements, so be sure to explain your decisions).
- Note: Your submission should be explained and able to be compiled and run from just your submitted files in your final submission. This means that you need to include any files provided to you that are necessary for your project to compile or run.
- Note: Source code and related documents submitted must be designed and implemented by the student submitting the work and any code must compile and run on one of the instructor's machines in order to be graded. (to create a working program quickly: get it working simply, then add to it; if at some point it stops compiling you will better know where an error was introduced)

5.3.2 **Final**: Final project – 50%

- There will be a final project to test the overall ability to understand, design, implement, test, and reflect on the problem solving and programming knowledge and skills covered in the class.
- The final project will be a mix of in-class (initial design, testing, and implementation discussion) and take-home (final design, testing, and implementation) elements.

5.3.3 Final grades will be given out based on the following based on score in the class:

90-100%: A
80-89%: B
70-79%: C
60-69%: D
00-59%: F

5.4 Reminder: A passing grade in order to count for course requirements for CS classes is generally a C or above.

6. Other Administrative Information:

6.1 For a list of general administration information (note that this list is not intended to be exhaustive), such as:

- contacting me,
- accessibility resources,
- expectations of student conduct,
- communications,
- student assistance,
- miscellany,
- nondiscrimination & nonharrasment,

(each section contains a number of sub-sections and is not meant to be exhaustive of all situations) see my administrative information document: *administrative information* document.