CS 262: Introduction to Low-Level Programming
Course Syllabus - Fall 2020 (08/24/2020 – 12/16/2020)
3 credits

George Mason University
Department of Computer Science

Format: Asynchronous
Class Location: Online (Blackboard Collaborate Ultra)

Sections: 001, 002
Instructor: Prof. Ana Loreto Gonzalez
Email: loreto@gmu.edu
Office Hours: MW 10-11AM using Blackboard Collaborate Ultra

Sections: 004, 005, G01
Instructor: Prof. Ping Deng
Email: pideng@gmu.edu
Office Hours: MW 3-4PM using Blackboard Collaborate Ultra

Textbook

- Brian W. Kernighan and Dennis M. Ritchie, The C Programming Language, 2nd ed., Prentice Hall, 1988

Complementary Books:

- Byron S. Gottfried, Programming with C, 2nd ed., Schumm's Outline, 1996 or the latest
- Peter Printz and Tony Crawford, C in a Nutshell: A Desktop Quick Reference, 1st ed., O'Reilly', 2006
- David Griffiths and Dawn Griffiths, Head First C, 1st ed., O'Reilly', 2012

Course Description

This course is intended to prepare students for topics in systems programming. It emphasizes relevant concepts of the C programming language, as well as the use of main commands of the Unix Operating System.

C is a high-level programing language that offers the programmer direct access to much of the underlying hardware and direct access to some operating system services for programs running under Unix. These features make C the preference language of choice for system programming.

Prerequisites: (CS 110* or 101*) and (CS 211 or 222) >> *May be taken concurrently

Course Outcomes

1. Be able to implement, test and debug a designed solution to a problem in a low-level programming language, specifically the C programming language.
2. Demonstrate a good understanding of C language constructs such as pointers, dynamic memory management, and address arithmetic.
3. Demonstrate a good understanding of C libraries for input and output, and the interface between C programs and the UNIX operating system.
4. Demonstrate an ability to use UNIX tools for program development and debugging.

Course Topics

The course will cover the following topics in no particular order:

- C Types, Operators, and Expressions
- Basic I/O, Input and Output Libraries
- File I/O
- Control Flow
- Functions and Program Structure
- Strings
- Pointers and Arrays
- Dynamic memory allocation
- Structures
- Bitwise operations
- The Unix System Interface
- vi/vim
- Debugging using GDB and Valgrind
- Compiling, Linking, Makefiles, using multiple source files

Evaluation and Grading

Grade Distribution

- Lab assignments 20%
- Projects 30%
- Quizzes 15%
- Midterm exam 15%
- Final exam 20%

*To receive a passing grade in this course, the average score for your two exams **MUST** be at least 60% **AND** you must submit all projects and labs*

Note: Lab attendance is not mandatory but has up to 2% bonus credit if attend at least 10 labs

Letter Grade Distribution

Your overall course score, S, will be the sum of these points.

S >= 98	A+
S >= 90	A
S >=88	B+
S >=80	B
S >=78	C+
S >=70	C
S >=60	D
S <60	F

Grading Elements Policies

- **Lab assignments:** Description for lab assignments will be posted on Blackboard. Submissions must be through Blackboard by the due date. Labs are intended to clarify the required aspects for lab assignments. During labs GTAs and UTAs will provide assistance and give hints to develop your programs.

  Late lab assignments will automatically be assessed a 50% penalty.

- **Projects:** Programming projects will be posted on Blackboard and student's solutions must be submitted on Blackboard by the assigned due date. If your program is incomplete, you may still submit it but your code must run without obvious errors (even if all functionality is not present). Notice your GTA relies on running your program as part of your grade determination. Accordingly, any programming assignment that is submitted but does not compile will receive no more than 25% credit. A programming assignment submission that has major errors when it is run will receive no more than 50% credit.

  Late projects will be penalized 10 points per day (incl. weekend days/holidays) for the first five days past the due date. After that time, late projects will be penalized 5 points per day for the next five days. The maximum late penalty will be 75 points.

The cutoff for on-time submission is 11:59 pm on the due date

If your program isn't the way you'd like it to be when the deadline is near, you could submit it. The system permits you to retrieve and resubmit your assignment until the due date, so you may resubmit if you improve your program prior to the deadline. The last submission is the one graded by your GTA. No resubmissions may be made after a project has been graded.

*All projects and lab assignments **must be submitted no later than the last day of regular classes** for the semester in order to earn a passing grade in the course (unless previous arrangements have been made with your instructor).*

- **Quizzes and Exams:** All quizzes and exams must be taken on the scheduled date/time. Passing this time, NO make-up of exams or quizzes are given unless previously arranged with the instructor.

Important Information

- **Class Communications:** CS 262 will be using Piazza and Blackboard for most class communications. You are responsible for any notifications or information posted on Blackboard/Piazza either by your instructor, your GTA or the class UTA(s), and you will need to check the systems regularly for such notices.
  Individual communications with the professor/GTA/UTA may be done by email using your GMU email account. When you email, please try to include your name, the class number and the topic in the subject header.

- **Special Accommodations:** If you are a student with a disability, please see your instructor and contact the Office of Disability Services (ODS) at (703) 993-2474. All academic accommodations must be arranged through the ODS: http://ods.gmu.edu/
- **Honor Code Policies:** All students are expected to abide by the GMU Honor Code. This policy is rigorously enforced. All class-related assignments are considered individual efforts unless explicitly expressed otherwise (in writing). Review the university honor code and present any questions regarding the policies to instructor.

Cheating on any assignment will be prosecuted and result in a notification of the Honor Committee as outlined in the GMU Honor Code. Sharing, collaboration, or looking at any code related to programming assignments that is not your own is considered cheating. See Programming Polices below.

The computer science department has an additional, more restrictive CS Honor Code that you are also subject to. Make sure you read and familiarize yourself with these rules.

Programming Policies

(1) **No sharing or discussion of code for assignments.** Unless specifically stated otherwise, all assignments are individual assignments, not group assignments. Students are expected to do their own work, not to share programs with each other, nor copy programs from anyone else. However, you may offer limited assistance to your fellow students regarding questions or misunderstandings on their programming assignments. Suspected honor code violations are taken very seriously, and will be reported to the Honor Committee. (See CS Honor Code)

(2) **No incorporation of code from any source external to the course.** You may not incorporate code written by others. Of course, you may freely use any code provided as part of the project specifications, and you need not credit the source. Working something out together with an instructor or GTA will not require crediting the source.

(3) **Back up your program regularly.** You are expected to back up your program in separate files as you get different pieces working. Failure to do this may result in your getting a much lower grade on a program if last minute problems occur. (Accidently deleting your program, having problems connecting, etc., will not be accepted as excuses.)

(4) **Keep an untouched copy of your final code submission.** It is important that you don't touch your programs once you have made your final submission. If there are any submission problems, consideration for credit will only be given if it can be verified that the programs were not changed after being submitted.

(5) **Code must run on Mason gcc.** Students may develop programs using any computer system they have available. However, submitted programs must run under gcc compiler available on Mason. Your documentation should clearly state which software was used for compilation, and once makefiles are introduced, a makefile should be included with each assignment submission.