CS 262: Introduction to Low-Level Programming
Course Syllabus – Spring 2025
3 credits

George Mason University
Department of Computer Science

Course Basics

Sections: 002, 005
Professor: Hamza Mughal
Email: hmughal2@gmu.edu
Office Hours: 1:30 pm – 2:30 pm M/W
Office: BUCHAN D217F

Sections: 004
Professor: John Otten
Email: jotten2@gmu.edu
Office Hours: 12:00 pm – 1:00 pm M/W
Office: ENGR 5335

Sections: 006
Professor: Tamara Maddox
Email: tmaddox@gmu.edu
Office Hours: 3:00 pm – 4:00 pm M/W
Office: ENGR 5347

Sections: 007
Professor: Bobby Chab
Email: rchab@gmu.edu
Office Hours: T/TR 3:00 pm – 4:00 pm
Office: TBD

Lectures:

Section	Days	Time	Building	Room
002	T,Th	12:00 pm – 01:15 pm	Innovation Hall	103
004	M,W	01:30 pm – 02:45 pm	Enterprise Hall	80
005	M,W	03:00 pm – 04:15 pm	Innovation Hall	103
006	M,W	09:00 am – 10:15 am	Horizon Hall	2017
007	M,W	10:30 am – 11:45 am	Horizon Hall	2009

Textbook (Required)

- Kernighan and 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
- Printz and Crawford, C in a Nutshell: A Desktop Quick Reference, 1st ed., O'Reilly', 2006
- Griffiths and Griffiths, Head First C, 1st ed., O'Reilly', 2012

Course Information

Prerequisites: C or better in CS 211 or CS 222, and C or better in CS 110
* (CS 110 can be a co-requisite with CS 262)

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. This is a course on "low-level" programming using C which is taught with an emphasis on operations with pointers.

Most high-level programming languages (and particularly Java) insulate the programmer from the realities of the hardware on which the programs will run. C is the exception since it was originally designed to implement the Unix operating system. C offers the programmer direct access to much of the underlying hardware and, for programs running under Unix, direct access to operating system services. For these reasons C remains the language of choice for systems programming.

Learning 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 the ability to use UNIX tools for program development and debugging.

Course Topics

The course plans to cover the following topics (in no particular order):

- C Types, Operators, and Expressions
- Basic I/O, Input and Output Libraries
- Control Flow
- Functions and Program Structure
- Strings
- Pointers and Arrays
- Dynamic memory allocation

- File I/O
- Structures
- Bitwise operations
- Multiple source files
- The Unix System Interface
- vi/vim
- Debugging using GDB and Valgrind
- Compiling, Linking, and Makefiles

Evaluation and Grading

Category	Percent	Notes
Lab assignments	10%	1 lowest lab dropped
Projects	30%	
Quizzes	20%	1 lowest quiz dropped
Midterm exam	15%	
Final exam	25%	Cumulative

Course Work

The latest you can turn in work is 48 hours after the posted deadline, after this time the submission link will be not available anymore.

- **Lab assignments:** Description for lab assignments will be posted on Blackboard. During Lab recitations, the lab instructor and one or more UTAs will be available to help students with the assignment. Submissions must be through Blackboard by the due date.

- **Projects**: There will be 3 projects in the semester. Project descriptions 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 for partial credit. However, 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.
  - The cutoff for on-time submission is 11:59 pm on the due date.
  - Note: Email submissions are not accepted.

Students are responsible for verifying that all submissions are the correct file and that the submitted files can be extracted correctly. Please note that once a submission link expires, we do not accept resubmissions due to corrupted files or incorrect file submissions.

Late Work

Late projects and labs are penalized 10% per day (incl. weekend days/holidays). For instance, if a lab assignment score is 100 points and the submission is one day after the due date, 10% will be deducted, and if the submission is two days after due, 20% is deducted.

If your program isn't the way you'd like it to be when the deadline is near, submit it anyway for

partial credit. 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.

Each student gets **THREE Emergency-Day tokens.** These tokens are automatically used by submissions that are between 0-24 and 24-48 hours late to avoid points penalty. These late tokens are only applicable for project assignments.

It is highly recommended to attend your lab sessions, as they are intended to clarify the necessary aspects for lab and project assignments. During labs, GTAs and UTAs will provide assistance and give hints to develop your programs.

For each lab that you attend in person, **you will earn up to 0.1% extra credit up to a maximum of 1% extra credi**t added to your final course grade.

Quizzes must be taken on the scheduled date/time. Quizzes may not be made up unless due to extenuating circumstances, such as illness, etc.

Exams must be taken on the scheduled date/time.

If you know in advance that you are unable to take an exam on the scheduled date for a valid and unavoidable reason, you must notify your professor at least one week before the scheduled exam date to make arrangements.

Replacement Midterm policy: If you perform better on the final exam than on your midterm, the midterm grade is replaced with the final exam grade.

Per departmental policy, you must pass a significant exam threshold to receive a passing grade in this class regardless of your performance on other assignments. To receive a passing grade in this course, your final exam grade **MUST** be >= 60% or the average exam grade **MUST** be >=65%. For the replacement policy to take effect, you must take the midterm exam.

If you feel points have been incorrectly deducted, contact the grader. For all projects and lab assignments, that is your GTA. For exams, that is your professor. Contesting of grades on any/all submissions must be requested within one week of receiving the grade. No grade changes will be considered after that deadline.

There is no compensation for a missed assignment. Only if the circumstances that caused you to miss an assignment are justifiable (e.g., surgery), contact the professor no later than 3 natural days after the due date and attach supporting documentation, otherwise your case will be dismissed.

Class Communications

CS 262 will be using Piazza, Blackboard, and email for most class communications. You are responsible for any notifications or information posted on Blackboard/Piazza on via email either by your instructor, your GTA or the class UTA(s), and you will need to check the systems regularly for such notices. Some information may be disseminated through these systems rather than in class.

Privacy and FERPA

Students must use their Mason email account to receive important University information, including communications related to this class. The instructor and GTAs cannot respond to messages sent from or send messages to a non-Mason email address.

Video recordings of class meetings that are shared only with the instructors and students officially enrolled in a class do not violate FERPA or any other privacy expectation. All course materials posted to Blackboard, or any other course site are private; by federal law, any materials that identify specific students (via their name, voice, or image) must not be shared with anyone not enrolled in this class.

For email communication, use your Mason email, including the subject and your section. Please note that emails sent on weekends may not be answered at that time and will be reviewed until Monday and answered in the order they are received.

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/

Inclusion

Every student in this class, regardless of background, sex, gender, race, ethnicity, class, political affiliation, physical or mental ability, veteran status, nationality, or any other identity category, is an equal member of our class. If you encounter any barriers to your inclusion, please feel free to contact your professor.

Academic Standards

GMU is an Academic Integrity standards university; please see the Office for Academic Integrity for a full description of the code and the honor committee process, and the Computer Science Departments Honor Code Policies regarding programming assignments. The principle of academic integrity is taken very seriously, and violations are treated gravely. What does academic integrity mean in this course? When you are responsible for a task, you will perform that task. All class-related assignments are considered individual efforts unless explicitly expressed otherwise (in writing).

Cheating on any assignment will be prosecuted and result in a notification of the Honor Committee as outlined in the GMU Academic Integrity standards. Sharing, collaboration, or looking at any code related to programming assignments that is not your own is considered cheating. Any attempts at copying or sharing code, algorithms, or other violations of the Academic Integrity standards simply will not be tolerated. We use automated software to flag suspicious cases, and then review them to find the cases that must be submitted to the Office of Academic Integrity with recommendation to fail the course (F) plus further measures.

Another aspect of academic integrity is the free play of ideas. Vigorous discussion and debate are encouraged in this course, with the firm expectation that all aspects of the class will be conducted with civility and respect for differing ideas, perspectives, and traditions. When in doubt (of any

kind) please ask for guidance and clarification.

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 Academic Integrity standards 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 compile and run on Zeus.** Students may develop programs using any computer system they have available. However, submitted programs must work under GCC compiler available on Zeus. 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. Programs that do not compile or are not runnable cannot earn more than a 50%.

Using ChatGPT (or any other LLM)

ChatGPT or other LLM models may not be used in this course as an assistant in the projects or other assignments unless otherwise specifically stated by the instructors.

Students who replace their own learning and assignment work with materials prepared by Generative-AI models:

• Sacrifice the opportunity to acquire the knowledge, skills, and critical thinking taught in the course.
• Risk being unable to perform to expectations in the academic environment when Generative-AI models are unavailable, such as in exams
• Ultimately endanger their employability if they are unable to produce work other than that produced by Generative-AI models