CS 262 Spring 2023 - Introduction to Low-Level Programming
George Mason University
Department of Computer Science

Course Basics

Instructor: Hamza Mughal
Email: hmughal2@gmu.edu
Office: BUCHAN D217F
Office Hours: TBA
Section 02 (T/TR 1:30 PM - 2:45 PM in East 201), 03 (M/W 12:00 PM - 1:15 PM in Planetary Hall 129), 04 (3:00 PM - 4:15 PM in Planetary Hall 129)

GTAs: Farina Faiz, Minyoung Kim, Mohamed Aghzal, Nate Krasner, Amirreza Hajrasouliha, Wenjie Xi, A H M Rezual Karim
UTAs: Oluwafunmiso Adeniyi, Nate Peterkin, Hailey Lee, Nicholas Bacon, Jeffrey Nguyen, Nitin Krishna Aluru, Wilson Tran, Samantha Amancio, Stephen Happ, Dheer Tammina, Saurabh Basak, Rishikesh Pisharody, Catherine Nguyen, Lloyd Amaranto, Brandon Landers

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)

Readings

Kernighan and Ritchie, The C Programming Language, 2nd ed., Prentice Hall, 1988 (required)

Byron S. Gottfried, Programming with C, 2nd ed., Schumm's Outline, 1996 or the latest (complementary)

Peter Printz and Tony Crawford, C in a Nutshell: A Desktop Quick Reference, 1st ed., O'Reilly, 2006 (complementary)

David Griffiths and Dawn Griffiths, Head First C, 1st ed., O'Reilly, 2012 (complementary)

Course Description

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.

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.

Learning Outcomes

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

Course Topics

This is a course on "low-level" programming using C. We will learn C with heavy emphasis on pointer operations.

List of Planned 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
Structures
The Unix System Interface
vi/vim
Bitwise operations
Makefiles, Compiling, and Linking
Debugging using gdb and Valgrind

File I/O
Multiple source files

Grading

20% Lab Assignments (11 labs, lowest lab dropped, each lab worth 2%)
30% Projects (3 programming assignments, each assignment worth 10%)
10% Quizzes (6 quizzes, lowest quiz dropped, each quiz worth 2%)
20% Midterm
20% Final (cumulative)

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%

If you perform better on the final exam than on your midterm, the midterm grade is replaced with the final exam grade. 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 instructor at least one week before the scheduled exam date to make arrangements for a make up

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

Grade Cutoffs

A+ [98-100] A [92-98) A- [90-92)
B+ [88-90) B [82-88) B- [80-82)
C+ [78-80) C [72-78) C- [70-72)
D [60-70)
F [0-60)

Labs

A short programming assignment will be given at the beginning of each lab. The lab instructor and one or more UTAs will be available to help students with the assignment. If not completed, the lab may be taken home. Lab assignments will be due either at the designated time for submission on Blackboard or (if no submission is given on Blackboard) at the beginning of the following lab period. No late lab assignments will be accepted.

Projects

In addition to the labs, there will be multiple larger programming projects. Programming assignments will be posted on Blackboard as they are assigned and must be submitted on Blackboard on the assigned due date by 11:59 pm. If your program is incomplete, you may still submit it for partial credit. However, your code must run without obvious errors (even if all functionality is not present). Building your programs in a modular fashion and debugging as you go are crucial concepts in program development. Additionally, your TA relies on running your program as part of your grade determination. Accordingly, any programming assignment that is submitted but either does not compile or has major errors when it is run will receive no more than 50% credit. Email submissions for projects 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 will be penalized 10% per day (incl. weekend days/holidays). You should recognize that this can cause major penalties for incomplete programs, so start work early! If your program isn't the way you'd like it to be when the deadline is near, submit it anyway for partial credit. In fact, submit early and often! 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 latest you can turn in work is 48 hours after the deadline

Students begin the semester with two One-Day-Late tokens. If you have a token left and turn in work late, the token is spent and no other deduction is made on the work. The late tokens may only be applied to projects and not labs. An unused late token is worht 0.5% extra credit at the end of the semester

Quizzes

Quizzes are in person. Quizzes must be taken on the scheduled date/time. No make-up quizzes are given.

Exams

Exams are in person. 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 instructor at least one week before the scheduled exam date to make arrangements for a make-up

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. Some information may be disseminated through these systems rather than in class. Individual communications with the instructor/GTA/UTA may be done by via Piazza utilizing private posts

Special Accomodations

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

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 can not respond to messages sent from or send messages to a non-Mason email address. We will not list your Mason email address on any public forum or provide it to any other students. Your Mason email address will be provided to grading staff (GTA and graders). If this is an issue, please contact the instructor so that we can figure out another option. 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

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 contact your professor

Honor Code

GMU is an Honor Code 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? Essentially this: 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 Honor Code. 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 honor code 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 a recomendation to fail the course plus further measures. The penalty for cheating will always be far worse than a zero grade, to ensure it's not worth taking the chance. 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

No sharing or discussion of code. Unless specifically stated otherwise, all assignments are individual projects, not group projects. Students are expected to do their own work, not to share programs with each other, nor copy programs from anyone else. This means you may not discuss program design or strategize solutions with anyone except your instructor or a course UTA or GTA. However, you may offer more limited assistance to your fellow students regarding specific questions on their programming assignments by responding to queries on Piazza. Any sharing of code or discussion of programming projects, except within the parameters of Blackboard, constitutes an honor code violation. Suspected honor code violations are taken very seriously, and will be reported to the Honor Committee. Read the GMU Honor Code and CS Department Honor Code. You are bound by these codes.

No incorporation of code from any source external to the course. You may not incorporate code written by others, such as code found on the Internet or any of the numerous CS books available. You may freely use any code provided as part of the project specifications, without any need for crediting the source. However, if you use code provided by your instructor (other than that given as part of the project specifications) or from the course textbook, you must document what portion came from those sources.

Piazza. We encourage the use of Piazza to discuss assignments and assist one another with programming questions. You may ask questions or respond to queries on Piazza regarding projects or other assignments, so long as you do not post any C code or detailed pseudocode, and so long as you do not provide specific solutions to the overall problem or algorithm design (even in English). Often, students believe that "simple" code is acceptable to place on Piazza. However, because there is a wide variation in what different students and instructors regard as "simple," we must be very strict about the ban against Piazza code. Only an instructor, GTA or UTA is permitted to place code on Piazza unless it is code that has already been provided to all students (either as part of the assignment specification itself or within the class textbook).

Discussing Piazza postings outside of Piazza. Please note that Piazza assistance must remain on the forum. "Summarizing" Piazza statements or responses to another student verbally regarding an assignment is *not* acceptable, and is subject to the above ban on discussing assignment solutions. While it may seem harmless, Piazza was set up so that all assistance could be overseen by instructors/TAs/UTAs, and it is nearly impossible to truly duplicate Blackboard/Piazza discussion outside of the actual forum, thus creating the potential for either (unknowing) mistaken advice, or for unfair advantage by certain students. If you truly wish to assist a fellow student, encourage him or her to log onto Piazza, and direct him/her to specific postings you find helpful.

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. (Accidentally deleting your program, having problems connecting, etc., will not be accepted as excuses.)

Keep an untouched copy of your final code submission. It is important that you not 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.

Code must run on Zeus using gcc. Students may develop programs using any computer system they have available. Please note, however, that submitted projects must run under the 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. No extensions will be given due to compiler incompatibilities.

Electronic Devices

Please feel free to use laptops, tablets, etc. to take notes, review slides, etc. during class. If you know that you will need to be doing other things on your screen/device (answering emails, perusing tiktok etc), please sit in the back row to ensure that other students are not distracted by your screen. We all need to do this stuff some times, and I have no judgements on you if you choose to sit in back with your screen open - just do not distract those sitting around you.

In Class Activities

Most lectures will feature interactive activities and/or quizzes. These quizzes and activities are meant primarily to help me understand how well you (and the class as a whole) are understanding the material.

You are strongly encouraged to bring your laptop or phone to class so that you can participate in the activities.