Computer Systems

CAS CS210 - Fall 2022

https://www.cs.bu.edu/courses/cs210/

https://piazza.com/class/l6ouw7eaofo51w

https://learn.bu.edu/

https://jupyterhub-redhat-ods-applications.apps.buaws-prod2.ysdy.p1.openshiftapps.com/

https://www.gradescope.com/courses/435472

Lectures: Tuesday and Thursday 9:30AM-10:45AM LAW AUD

Discussions: Monday 8:00-8:50AM, 9:05-9:55AM, 10:10-11:00AM, 11:15AM-12:05PM, 12:20-1:10PM, 1:25-2:15PM, 2:30-3:20PM, 3:35-4:25PM KCB 103

CS210 Staff Fall 2022

	Instructor	TF	TF	TA
Name	Professor Appavoo (he, him, his)	Muhammad Faisal (pronouns)	Yuanli Wang (pronouns)	Ke Li (pronouns)
Email	jappavoo	mfaisal	yuanliw	keli2023
Office	MCS 227E	EMA302	EMA302	EMA302
Office Hours	Tue 5-7PM,Thu 12-2PM	Wed 8-9:30AM	Wed 3:45-5:15PM	Fri 10-12PM

Course Assistance (CA's): We are fortunate to have a number of undergraduate course assistant (CA's) as members of the course staff. They will be assisting you in the labs and holding office hours each week. Below are their names and contact info.

Office Hours Location: EMA302 RM 302, 730 Commonwealth Ave.

Name	Email	Office hours
Kamila Uranayeva (she/her)	ukamila	Wed 11-12:30AM
Jason Lee (he/him)	jaslee20	Wed 2:30-4PM
Alexis (Lexi) Pfalzgraf (she/her/hers)	alexispf	Wed 1-2:30PM
Leanne Tong (she/her)	leannet	Fri 9-10:30AM
CJ Parra (they/them)	acparra	Thu 4-5:30PM
Brian Tao (he/him)	briantao	Thu 10:30-12PM
Gwen Liu (she/they)	gwenl	Tue 2:30-4PM
Jake Gustin (he/him)	gustinj	Fri 8-9:30AM

Midterms Fall 2022

You must be able to attend the two 75 minute midterms that will be held on the evenings of **Oct 6, 2022 at 6:45PM - 8:00PM (Midterm I)** and **Nov 10, 2022 at 6:45 - 8:00PM (Midterm II)**. Note we do not normally meet during these times.

Your first written assignment is to read, sign, date (see page 15), and submit a copy of this syllabus to gradescope. See PS0 on Gradescope and Piazza for details.

Have Fun!

We love this material and hope you will have fun learning about how computers actually work. The rest of this document provides further details, including a tentative lecture by lecture breakdown.

Here's to the adventure that we are about to embark on. We will start going down a road that will take us from being users of computers to becoming masters of the modern digital world!

To help you understand our approach to this class the following discusses our pedagogy. It also includes some advice for your success and can help you understand our expectations.

Pedagogy: Teaching Method and Practice

The material of this class is designed for you to gain a solid understanding of the concepts and mechanisms for how computer systems work. This knowledge is the foundation for you to more deeply understand, and put in context, everything else you will study or construct as a computer scientist.

This is not a class in which you will be given recipes, taught procedures for doing things, or solely taught syntax. We will, rather, focus on teaching you concepts and principles that underly all computer systems. As part of this we will teach you the attendant skills that you need to explore and solidify these concepts. These skill will distinguish you as professionals that know how things work and who known what they are doing. We want you to become self-directed, knowledgeable computer professionals, whose creativity can be translated into creations.

We don't want you to be programmers destined to be replaced by ML code generators. Developers, today, that lack a grasp of the fundamental principles that underly the tools they use can easily become obsolete. Their skills become irrelevant when the particular tools and languages they have been taught are replaced.

The lectures, discussions, text and especially the assignments are designed to get you thinking and exploring. They are not exercises for you to reenforce a particular mechanical procedure. They, rather, encourage you to "poke around" and engage in self-discovery. Through the act of "doing" you can concretize the concepts and ideas we discuss. Do not, however, assume that this means any less of a time commitment. If anything you need to allow yourself extra time so you can go down rabbit holes and following a trail of manual pages and code that relate to the ideas.

Comments about programming and assignments

We are programmers too! The whole point of computer systems is to enable programmers. To understand a computer system means understanding programming. Our goal in this class, with respect to programming, is some what different that the usual one of learning to program or learning a specific programming language. We assume you already know the basics of programming both procedural and object oriented. In this class are seeking to understand: 1) the low-level programming model of the hardware and 2) the environment provided by the operating system to execute programs. All other programming models and environments, including those that you are already familiar with, are built on top of these.

While we will be programming, our goal is not to instruct you in a particular syntax or teach you the nuances of assembly or C programming. Our aim is for you to understand how the software and hardware work and interact, by learning the core mechanisms and ideas of assembly and C.

Our focus will be on exposing and understanding the foundational programming environments of the hardware and Operating System.

So programming in this class is both a tool and skill for engaging with computer systems. You will need to program to fully engage in the kind of exploration that can teach you to be a master of the digital universe. You will need to explore the syntax and mechanism of assembly and C so that you to read and write programs that let you explore the hardware and OS. This can mean hours of reading manual pages and writing and debugging programs that aren't even part of an assignment. These exploratory sessions will help you understand the concept and mechanism that a particular assignment is about. Learning to explore software and hardware, in this way, is a critical skill that will ensure your future success in the field of applied computer science. Remember, any artist that truly understands their medium must spend hours engaging with it.

Assignments

While it seem odd, do not expect to understand everything about an assignment, or what you need to do. When we give you an assignment we will purposefully leave aspects of the assignment vague and omit a recipe on how to complete it. To create or find a solution you will often need to use the tools and skills we cover to discover what you need to know. This can include reading test code we provide, writing your own tests, and exploring manual pages. With this in mind, when you feel lost or frustrated, know that this is normal. Do not immediately post a question or search the internet seeking a clear "instruction" or set of steps to follow. Spend some time exploring. If you don't know where or how to begin exploring then by all means ask for advice. We want to have those conversations with you. Discussions will often be structured to help bootstrap the exploration process.

Further, do NOT program to the tests we provide you to evaluate the an assignment. That is to say, "do whatever it takes and only whatever it takes" to pass any provided tests for a programming assignment, and do this in the least amount of time and effort. With this in mind students treat the problems of an assignment like black-boxes. They avoid exploring the test code and related topics. Students often expect to follow the instructions of an assignment, like a recipe, that if blindly followed, will produce a solution.

If a test passes, they consider themselves done, and if it fails, they get frustrated and simply keep randomly trying to modify their code until the test passes. This is a very bad idea in this class. Tests we provide are there to get you thinking and exploring. You should get in the habit of reading any provided test code and even modifying and extending it.

Understand what our code is testing, how it works, and use the debugger. You can learn a lot from our test code itself. You can and should use it and the debugger to write your own additional tests. This way the process of getting a solution is not a random process but rather one in which you learn and hone your skills.

Working hard to understand and complete the assignments will help you gain the knowledge of this class (and make the exams easier).

Course UNIX environment

We have carefully constructed a web-browser accessible UNIX Terminal oriented environment for you. This environment is "real". All the software in it is the same software used industrially. Everything that we practically do is relevant and reflective of how things work in the "real world".

The environment, at first, may seem old and frustrating. Further, you might think it is irrelevant to how things work today. The opposite, however, is true. The environment is designed to let us strip away all the layers and expose to you how things "really" work. Learning to construct software with the underlying core tools and mechanism puts you on the road to being a "power" programmer. That is to say, one who understands what is at the bottom of all the layers and can work at even the lowest.

We find that many students, once they get comfortable working at this level, really enjoy the power, understanding and ability that comes with it. Regardless, you will then be able to make informed decisions about the tools you use and understand what they provide you and what they hide from you.

Understanding the concepts and skills we cover will help establish your credentials as computer scientists, who not only understand what goes into constructing software, but those who can participate in all aspects of the technologies that go into computer systems.

GDB

The software environment we have prepared includes several tools beyond those strictly needed to develop both assembly and C. Perhaps the most important of these is a machine oriented debugger called `gdb`. Unlike other debuggers you may have used it is a tool for you to do much more than just debug your code. It will, rather, let you freeze arbitrary running programs, examine the state of the hardware and manipulate it. The syntax of `gdb` can be a little difficult to get started with. It is, however, really in your best interest to learn to use `gdb` and not shy away from it. Historically, students who learn to use `gdb` have a far easier time with the assignments. These students, also tend to have an easier time understanding the connection between the conceptual and applied topics of this class.

Summary

So the bottom line is we have set up an opportunity for you to learn about computer systems by poking, prodding and writing low-level code. Please exploit this opportunity.

At first this approach might seem odd and you might be wondering why we just don't give you the "answer", or simply "tell" you how things work. But the act of exploration and "doing" provides your brain the opportunity to develop its own "deep" understanding. This takes time and engagement with the material. The magical world of computers systems are set of fascinating and beautiful ideas that have been brought to life in machines. It takes time and effort to be able to see and appreciate this beauty please give yourself the opportunity to do so.

Your curiosity and desire to know how things work are your best friends when it comes to this journey. Use them to motivate you to try things out and explore. Remember this will take time be sure to plan appropriately.

Course Description

Our goal is; 1) to provide a foundation for understanding how computer systems work from a software perspective, 2) demystify the complex layers of software that make up the world around us and 3) learn a new set of technical competencies.

The course roughly breaks down into 3 major parts as follows.

1. The UNIX Development Environment: Learning what an OS Kernel is, what User Process are and the tools of the trade that go into developing foundational software.
2. The von Neumann computer architecture: Learning the binary model of a computer and binary representation of software, through assembly programming.
3. Into the Light: Learning how we bridge the binary model and representation to a human friendly programming, through the "C" programming language tool-chain.

There is a companion set of online materials that are being developed that follows the above break down. While this materials is still under construction you are encouraged and welcome to use any portion.

CS 210 is a principal course for computer science majors. It provides the fundamental knowledge to understand what software and hardware are. It is also the background for courses in the systems area such as operating systems, compilers, networks, not to mention more advanced courses in computer architecture.

Prerequisites

This course assumes that students have a solid background in programming concepts from CAS CS 111. CS 112 is also recommended, but not essential for students with strong programming skills. CS 131 or MA 293 is important for the material on Boolean logic and data representation.

Course Materials

UCSLS Online: This material is under construction but we will refer to it where possible, "Under the Covers: The Secret Life of Software".

textbook: https://jappavoo.github.io/UndertheCovers/textbook

lecture notes: https://jappavoo.github.io/UndertheCovers/lecturenotes

lab manual: https://jappavoo.github.io/UndertheCovers/labmanual

CPAMA Text: K.N. King, "C Programming: A Modern Approach", Second Edition, W. W. Norton & Company, 2008.

Top Hat Pro: We will use the Top Hat Pro platform. Please be sure to have enrolled. Further information will be provided in class and on piazza.

Three other books you might find useful are:

Optional: R. Nigel Horspool, "C Programming in the Berkeley Unix Environment", 1987.

Optional: Brian W. Kernighan and Rob Pike, "The UNIX Programming Environment", Prentice Hall, 1984. (Another Classic Text).

CSAPP Text: Randal E. Bryant and David R. O Hallaron, http://csapp.cs.cmu.edu/, "Computer Systems: A Programmer's Perspective", 3rd. Prentice Hall, 2016, ISBN-13: 978-0-13-610804-7 (A text we used to use for this class)

Online Organization

1. Course website: https://www.cs.bu.edu/courses/cs210/
2. Piazza site: https://piazza.com/class/l6ouw7eaofo51w
   The piazza site will be our primary means of communication through the semester. This will include
   - posting announcements and updates to the weekly schedule
   - posting of class materials
   - posting links to assigned readings
   - messaging: A place to post questions and answers (do not use email) See details below.
3. Gradescope: https://www.gradescope.com/courses/435472
   We will use gradescope for submission and grading of assignments and exams. For programming portions of assignments we will exploit gradescope and github classroom (see below) integration. You must upload your assignments to gradescope from the matching github classroom repository.
4. Blackboard site: https://learn.bu.edu/
   While we have a blackboard site we will primarily use piazza and gradescope and reserve blackboard as a backup.
5. UNIX Development Environment: https://jupyterhub-redhat-ods-applications.apps.buaws-prod2.ysdy.p1.openshiftapps.com/
   As part of the online textbook material we will be using an online service to do all of our programming and exploratory work. You will need to follow the link and login with your BU credentials. See the UCSLS textbook for information about how to use the environment.
6. GitHub Classroom:
   We will be using git repositories for all assignments in this class. Each assignment will be a unique repository for you to manage and conduct your work. We will post an invitation link on piazza for each assignment that you will need to follow and accept. If you do not have github.com user you will need to create one. Additionally, you will need to do a one time registration to the github classroom. Doing the one time setup will be part the first assignment. Please post to piazza if you have any questions or difficulties.

Piazza

We have purchased and setup a course piazza site which can be accessed here:
https://piazza.com/class/l6ouw7eaofo51w.
The following is some guidance with respect to our use of piazza.
Piazza is a service we offer to ensure efficient and centralized communication.
During the semester we might make changes in the syllabus, schedule, or course policy. Changes will be posted on Piazza, and the information on Piazza will be considered to supercede the information on the Syllabus pdf.
Be sure to stay up to date with the information on Piazza!
Over the course of the semester the staff will use piazza to post:

- office hours information and changes to them
- links to reading
- links to lecture presentations/notes
- assignment materials and links
- discussion materials and links
- additional resources
- exam prep materials
- solutions to assignments and exams when appropriate

Piazza is also a place that you can post questions to the staff regarding:

Logistics: Lecture, discussion and office hour location and times.

Clarifications: If after having attended lectures and discussions you are still unclear what an question on an assignment means or what is expected of you please post.

Followups: If something was discussed in lectures, discussions or mentioned in the reading that you are would like to know more about please post.

Guidance: If there is a topic about computer systems that you would like guidance or more information about please ask.

Seeking help on Piazza and Guidelines

More generally you are welcome to post questions seeking help regarding the course material. Note however, this is not a public forum, we will moderate posts, removing or change the visibility as needed. Please keep the following guidelines in mind.

- Be polite and considerate.
- Always search to see if the topic of your questions has already been discussed.

- Piazza, is not a substitute for doing the readings, attending lectures, discussions, and office hours. If you notice a question whose answer can be found in these materials please politely direct the poster to the appropriate material.
- If you notice a questions that is a repeat please help you fellow student by directing them to the prior posts.
- If you notice a question that you can help with please post a response. If you are unsure that's ok, just acknowledge so, and be sure to ask the staff to clarify. A good way to learn, and solidify your own understanding, can be to try and explain something to others.
- Do not post questions that seek solutions to assigned problems nor should you provide such answers. In general please do not post code that relates to an assignment.
- If you are stuck:
  - Try and ask as specific a question as you can. Please don't simply post messages of the form "Help nothing works" or "I don't know what to do".
  - Explain what you have tried.
  - Describe, in as much detail as you can, what in the provided materials is confusing and does not make sense to you.
- If you have discovered something interesting that relates to the course by all means share it in a post.
- Piazza is not a support line to get your code/solutions debugged nor get answers to your assignments. Please refrain creating such posts.
- In general be willing to engage in conversations which are trying to help you understand versus just get the answer.
- Ask your questions early regarding assignments. **In general do not expect staff help within eight hours of a deadline.**
- Piazza is not a public forum to discuss concerns about the course or staff. Please arrange to meet with the professor regarding these concerns. If you are not comfortable discussing your concerns with the professor please contact the department chair.

COVID-19/Health Procedures

To promote a safe learning environment, students must adhere current University policies. At the time of writing the current policies can be found here:
https://www.bu.edu/dos/2022/08/10/important-health-information-fall-2022/.

Requirements and Grading

Midterms	30%	Average of the two midterm exams. The midterm exam average will be tentatively weighted 60% of the best grade and 40% of the lower grade.
Final Exam	30%	A final exam will be held during the assigned examination period. The exam will be cumulative covering all material from the course.
Assignments and Quizzes	30%	Several Assignments which can require both written and programming solutions
Participation	10%	See below.

To pass the course, you must earn a passing grade on each of the above components.

Grading

Grading (except for the final exam) is done by a number of class graders, under the direct supervision of the Teaching Fellow and the professors. If you have an issue with a grade (homework or exam), please contact the Teaching Fellow. Only if the issue is not resolved to your satisfaction, please contact the professor. Note the professor may opt to re-grade the entire submission. The professor's result will be the final grade assigned for the submission (note that this value maybe lower than the original score).

Grades must be appealed within one weeks of receipt.

Grading Scale

The final grades are *not* curved. The performance of the class as a whole is taken into account in assigning letter grades, but this can only improve your grade, not harm it.

Incompletes, Missed Work and Extensions

No incompletes will be given, except for reasons of dire illness shortly before the end of the course, and only if a significant amount of work has been completed (e.g., attending lectures, handing in most assignments, and attending the midterms).

Extensions and makeup exams will only be given in documented cases of serious illness or other emergencies. You cannot redo or complete extra work to improve your grade.

Bonus Work

Various assignments may have bonus components. You may work on these bonus component through out the semester. All bonuses will be due on the last day of classes. Seperate submission sites will be created for each bonus component. At the end of the semester a final bonus grade will be calculated. If you have met the standard grading requirements outline above, to pass the course, then the bonus grade will be added to your aggregate score. As such, the bonus components cannot help you pass the class but can improve a passing grade.

The combined bonus score will be included into your final grade as an additional assignment. For example if there are six regular assignments then each assignment will be worth 5 points towards your final grade. In this case your total bonus score can add at most 5 points to your final grade.

Collaboration Policy

You are strongly encouraged to collaborate with one another in studying the lecture materials and preparing for the exams. Problem sets will include:

- individual-only problems that you must complete on your own
- pair-optional problems that you may complete alone or with a partner.

For both types of problems, you may discuss ideas and approaches with others (provided that you acknowledge this in your solution), but such discussions should be kept at a high level, and should not involve actual details of the code or of other types of answers. You must complete the actual solutions on your own (or, in the case of a pair-optional problem, with your partner if you choose to use one).

Rules for working with a partner on pair-optional problems:

- You may not work with more than one partner on a given assignment. (However, you are welcome to switch partners between assignments.)
- You may not split up the work and complete it separately.
- You must work together (at the same computer or via a Zoom meeting) for all problems completed as a pair, and your work must be a collaborative effort.
- You and your partner must both submit the same solution to each problem that you did as a pair, and you must clearly indicate that you worked on the problem as a pair by putting your partner's name at the top of the file.
- For gradescope submission be sure to use the group submission option.

Academic Misconduct

We will assume that you understand BU's Academic Conduct Code:
http://www.bu.edu/academics/policies/academic-conduct-code

You should also carefully review the CS department's page on academic integrity:
http://www.bu.edu/cs/undergraduate/undergraduate-life/academic-integrity

Prohibited behaviors include:

- copying all or part of someone else's work, even if you subsequently modify it; this includes cases in which someone tells you what to write for your solution
- viewing all or part of someone else's work (with the exception of work that you and your partner do together on a pair-optional problem)

- showing all or part of your work to another student (with the exception of work that you and your partner do together on a pair-optional problem)
- consulting solutions from past semesters, or those found online or in books
- posting your work where others can view it (e.g., online)
- receiving assistance from others or collaborating with others during an exam, or consulting materials except those that are explicitly allowed.

Incidents of academic misconduct will be reported to the Academic Conduct Committee (ACC). The ACC may suspend/expel students found guilty of misconduct. At a minimum, students who engage in misconduct will have their final grade reduced by one letter grade (e.g., from a B to a C).

Midterms and Exam

There will be two midterm exams and one final exam which will include all material covered from the beginning of the semester until the day of the exam.

The two 75 minute midterms are held during the semester on:

Oct 6th and Nov 10th at 6:45PM till 8:00PM.

Note we usually do not meet in the evenings. It is your responsibility to ensure that you can attend the midterms. These dates are not flexible. The final will be held during the assigned exam slot. Please plan your work and travel plans accordingly.

Midterm and Exam Conduct

During exams you are not permitted to wear any hat with a brim, such as a baseball hat, that could obscure the adjudicator from seeing your eyes.

If you are not assigned a seat then we recommend that you seat yourself randomly and in a manner that ensures your work and that of your peers stays confidential.

If, during an exam, you are found consulting any other material than what is provided or specified, it will be considered as a possible case of academic misconduct. This includes using electronic devices and encompasses answering calls or messaging.

Again if you need to leave the room for any reason, including the restroom, you will be required to turn in your exam.

Assignments

A core aspect to understanding computer systems and becoming distinguished programmers comes from the doing. To this end doing the assignment is a critical aspect to this course. While it might be tempting to immediately search for code, don't do it! Please ask us instead. We are here to help you learn. The assignments are intended to engage you personally with the material, don't squander that opportunity and don't fall prey to plagiarism. Ask questions in the lecture, ask questions during discussions, ask questions during office hours, ... ask questions!

Schedule and Logistics

There will be seven assignments referred to as problem sets; PS0 to PS6. Each assignment is broken down into two parts; Part A written and Part B programming. Both parts will be provided to you as a private git repository that contains the files related to the assignment. Your solutions to both parts A and B will be submitted, graded and returned back to you on gradescope.

The first, PS0, is an introductory assignment to make sure that you are setup on all the course infrastructure and you will have one week to complete both parts.

The remaining six, PS1 to PS5 form the core assignments for the class. In general, something will be due every Monday at 11:59 PM, through out the semester. Both parts A and B of a problem set will be release, every other Tuesday's at 12:01 AM. Part A, the written component will generally be due on the following Monday at 11:59 PM. The programming portion, Part B will be due on the Monday after the written potion, again at 11:59 PM. Please consult the detailed weekly syllabus for the schedule. Carefully, read the Problem Set handout to clarify its due dates. If there are any changes to the assignment schedule we will post updates to on Piazza.

Any computer based work you need to do to complete either part A or part B of a problem set should be done in the provided online UNIX environment.

Each problem set will contain a `README.md` that will provide the general instructions for the assignment.

Written

Within a particular problem set you will find a pdf that forms part A of the problem set (the README.md will clarify which file is the pdf). This is the written component and you should provide all your answers in the space provided on the pdf.

You will need to download the pdf to your laptop and complete is as directed. You will submit your updated pdf to a gradescope submission site that we will create and post a link to.

PS0, part A, will walk you through what needs to be done for written assignments. In general, you can either update a copy of the pdf and submit that or you may print out the pdf and work on paper. If you choose to do the later you will need to scan, or take pictures, of your paper copy to upload to gradescope.

Programming

The repository for the the programing component will form part B of the problem set. To work on the programming portion you will need to create a local copy of the repository in the provided online UNIX environment. As you work on part B you may need to both update files and add new ones. All changes and new files must be "committed and pushed" back to your master repository. To do this you will need to use the appropriate git commands.

WARNING: The files and directories of the online UNIX environment are NOT permanent. If you disconnect or do not actively use your server it will be rebooted and all the files will be deleted. To permanently save your work you MUST frequently use git to "commit and push" copies back to your master git repository.

Part B of PS0 will help you get bootstrapped on what you generally will need to do for part B of problem sets.

A separate gradescope submission site will be created for part B. To this submission site you will need to submit a copy of your main git repository on github classroom . Don't forget to ensure that it is updated correctly with your latest version.

The programming portion of an assignment will be evaluated both with automatic and manual grading. For each part B we will provide you the same script that the autograder will run on your solution to test it. You should inspect this script to help you understand how your solution should work and what we expect. For the manual portion we will inspect your code and repositories.

Repository histories

Every time you git "commit and push" changes to your assignment repository a time stamped record is created. The "history" of your changes documents exactly when and what work you have done. We will inspect your histories to evaluate your effort and work. Please note if we do not observe a history that is indicative of a realistic effort, in a reasonable time frame, to produce a solution then we may assign you a zero for the auto graded portion of the grade.

With this in mind we recommend you start early and frequently commit and push your changes to your master repository. Remember, you do not have to have things in a working/solved state when you commit and push. Rather you should treat it more like the saving work and documenting what you have tried and where you are. So do it often!

Late Policy

Late problem sets: Problem sets must be submitted by the date and time listed on the assignment (typically by 11:59 p.m.). There will be a 10% deduction for submissions up to 24 hours late. We will not accept any homework that is more than 24 hours late. Plan your time carefully, and don't wait until the last minute so you will have time to ask questions and obtain assistance from the course staff.

Laptop Policy

Laptops: Students taking CS courses are expected to have a laptop capable of running a currently supported version of Microsoft Windows, Mac OS X, or Linux. See this page for more info:
https://www.bu.edu/cs/undergraduate/undergraduate-life/laptops

Office hours

The professor and TF's will hold office hours. The purpose of the office hours is to answer specific questions or clarify specific issues. Office hours are not to be used to fill you in on a class you skipped or to explain entire topics. Please come to class and to your discussion sessions.

To reach the teaching staff at times other than office hours, please use piazza.

Lectures

The topics of the lectures build on each other. You will find it very difficult with later topics if do not ensure understanding of a preceding topic. As such, we encourage you to reach out to the staff

during discussions and office hours to clarify your understanding. Engage in Piazza discussions. If you are still feeling lost send us a private message on piazza to find a time to chat.

Do not turn to the internet/social media, or any other sources if you are feeling overwhelmed. We are here to teach and help you master a subject that we love. We want to help you and see you succeed. There will be zero tolerance for plagiarism (see Academic Conduct above).

Each lecture will have pointers to readings. You should read this material prior to the lecture. In the syllabus **UC-SLS** refers to the online text "Under the Covers: The Secret Life of Software", and **CPAMA** refers to "C Programming: A Modern Approach".

Lectures, however, will not be restricted to text material or what is on online versions of the lecture slides. Lectures may cover additional or alternative material.

You will be responsible for all material covered in the lectures.

Participation and Quizzes

The participation portion of your grade will be based on your completion of the in-lecture quizzes and in-lecture questions, and on your consistent participation in the discussion sessions. You will receive full credit for participation if you answer at least 85% of the in-lecture questions and if you participate in at least 85% of the discussion sessions. If you complete x% of the questions or participate in x% of the discussion sessions for a value of x that is less than 85, you will get x/85 of the possible points.

Note that the above participation policy is designed to handle occasional absences due to illness or other special circumstances – including ones stemming from isolation for Covid. We will be recording the lectures and making the recordings available to everyone in the class. If you need to miss a lecture for any reason, you should simply watch the recording for that lecture as soon as possible after it is posted. **Please do not email your instructor for absences of this type.**

We will use the Tophat platform to conduct in lecture and discussion quizzes and gather attendance.

Teaching Fellows and Discussions

Students are expected to attend the weekly discussion section that they have been assigned to. Discussions are a critical component of this course, and attendance is mandatory.

In addition to the discussion the Teaching Fellows will hold office hours. See the first page of the syllabus for times and location.

A comment about syntax

Assembly language programming can feel very foreign at first. However, if you always clarify what each aspect of the syntax means, by exploring it with small programs of your own, that you run in the debugger, it will become much easier.

Warning: the syntax of the Java Programming language was derived from C, but as programming languages, the semantics of C and Java are very different. C is heavily influenced by the underlying resources and behavior of the real hardware of a computer, whereas Java was developed

around an abstract virtual machine model whose goal is to facilitate high-level programming and portability.

Be careful not to assume Java semantics and behavior when working with C. There are aspects of C, particularly pointers, explicit dynamic memory allocation, and formatted I/O, that do not exist in Java. The "C Programming: A Modern Approach" textbook is an approachable book that can be read cover-to-cover to obtain a good handle on the language.

Signature

name: _____

buid: _____

date: _____

Detailed Syllabus

The following is a rough weekly break down. The instructors will provide more details on piazza as the semester progresses.

Date	Activity/Topics	Readings	Assignments
Tue 09/06/22	Course overview and introduction.		PS0: OUT
Thu 09/08/22	UNIX Introduction and Preliminaries	ch1, ch2, ch3, 17.1, 4.1-4.6	
Mon 09/12/22	TBA		PS0: DUE
Tue 09/13/22	Programming, Files and Directories	4.7-4.11	PS1: OUT
Thu 09/15/22	I/O, Process Control and Credentials	ch5	
Mon 09/19/22	TBA		PS1A: DUE
Tue 09/20/22	Editors, Make and Terminal IDEs		
Thu 09/22/22	Version Control and GIT: The Basics		
Mon 09/26/22	TBA		PS1B: DUE
Tue 09/27/22	Assembly Programming Introduction	ch12, ch13	PS2: OUT
Thu 09/29/22	Writing some simple assembly programs	ch13, ch14, ch15	
Mon 10/03/22	TBA		PS2A: DUE
Tue 10/04/22	Assembly : Operations and Data Types	ch17	
Thu 10/06/22	Assembly : Program Anatomy I **Midterm I: 6:45 - 8:00PM - LOC: TBA**		
Mon 10/10/22	Indigenous People's Classes Suspended - No Discussions		PS2B: DUE
Tue 10/11/22	Substitute Monday Schedule of Classes Last Day to Drop Standard Courses (without a "W" grade)		PS3: OUT
Thu 10/13/22	Program Anatomy II : Functions		

Mon 10/17/22	TBA		PS3A: DUE
Tue 10/18/22	Program Anatomy III : Code as Data		
Thu 10/20/22	Program Anatomy IV: The Tree of Bytes and Data Structures		
Mon 10/24/22	TBA		PS3B: DUE
Tue 10/25/22	Assembly using the OS		PS4: OUT
Thu 10/27/22	Virtual Memory		
Mon 10/31/22	TBA		PS4A: DUE
Tue 11/01/22	Processor Caches		
Thu 11/03/22	In to the Light - C Intro		
Mon 11/07/22	TBA		PS4B: DUE
Tue 11/08/22	Using C to write and organize opcode bytes - Functions		PS5: OUT
Thu 11/10/22	Using C to map and organize data bytes - Data Types **Midterm II: 6:45 - 8:00PM - LOC: TBA**		
Mon 11/14/22	TBA Last Day to Drop Standard Courses (with a "W" grade)		PS5A: DUE
Tue 11/15/22	Using LibC to access the OS and escape the confines our process		
Thu 11/17/22	The rest of LibC and other libraries		
Mon 11/21/22	TBA		PS5B: DUE
Tue 11/22/22	Binary Programming		PS6: OUT
Wed 11/23/22	Thanksgiving Recess, Classes Suspended		

Thu 11/24/22	Thanksgiving Recess, Classes Suspended		
Fri 11/25/22	Thanksgiving Recess, Classes Suspended		
Sat 11/26/22	Thanksgiving Recess, Classes Suspended		
Sun 11/27/22	Thanksgiving Recess, Classes Suspended		
Mon 11/28/22	TBA		PS6A: DUE
Tue 11/29/22	Revisiting Integers: 2's complement		
Thu 12/01/22	Revisiting Data Types		
Mon 12/05/22	TBA		PS6B: DUE
Tue 12/06/22	Caching and Performance optimization		
Thu 12/08/22	TBA		
Mon 12/12/22	TBA Last Day of Classes		Last day for bonus submissions