Computer Systems

CAS CS210 - Fall 2023

website: https://cs-210-fall-2023.github.io/CS210-Website/
Piazza: https://piazza.com/bu/fall2023/cs210
UNIX: https://rhods-dashboard-redhat-ods-applications.apps.awscas210.zkf1.p1.openshiftapps.com/notebookController/spawner
Gradescope: https://www.gradescope.com/courses/572191
Lectures: Tuesday and Thursday 9:30AM-10:45AM CGS 129
Discussions: Monday: A2 8:00-8:50AM (CGS 523), A3 9:05-9:55AM (CGS 523), A4 10:10-11:00AM (CGS 421), A5 11:15AM-12:05PM (CGS 523), A6 12:20-1:10PM (CGS 321), B2 1:25-2:15PM (CGS 423), B3 2:30-3:20PM (PSY B39), B4 3:35-4:25PM (CGS 313)
B5 4:40-5:30PM (CGS 311), B6 8:00-8:50AM (PSY B35)

CS210 Staff Fall2023:

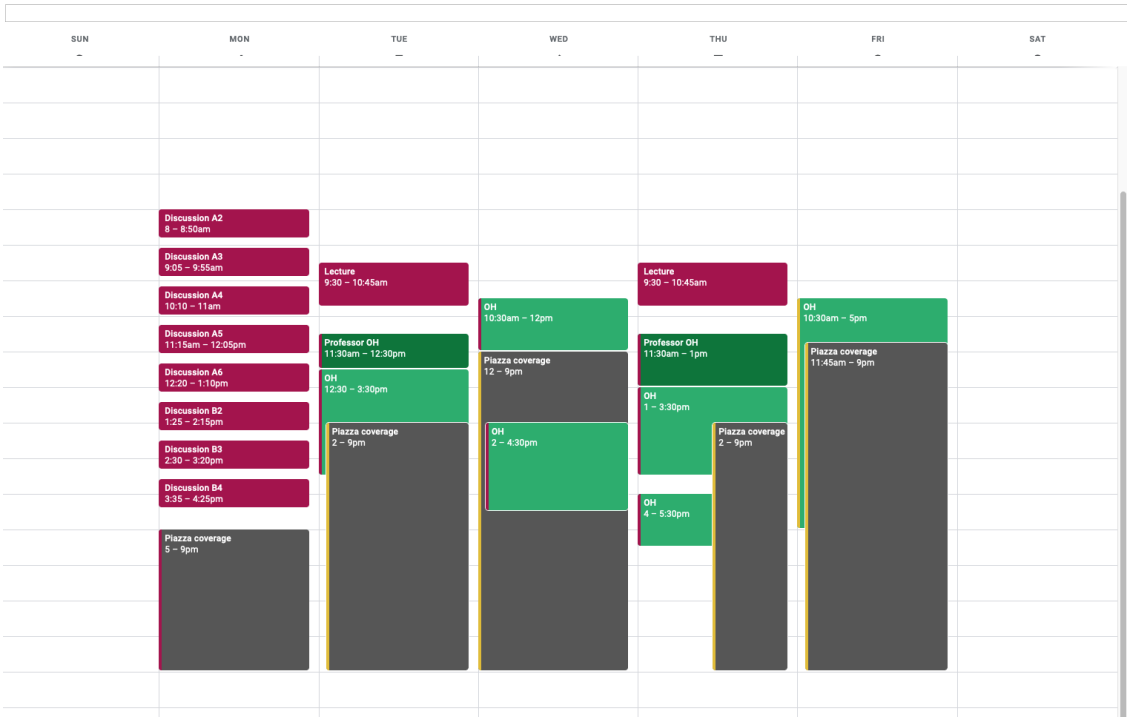
Role	Name	Pronouns	BU Email
Instructor	Vasiliki Kalavri	she/her	vkalavri
Instructor	Jim Cadden	he/him	jmcadden
Instructor	Preethi Narayanan	she/her	pnarayan
Course facilitator	Amy Feng	she/her	afeng99
Teaching Fellow	Teona Bagashvili	she/her	teona
Teaching Fellow	Papon, Tarikul Islam		papon
Teaching Assistant	Alexis (Lexi) Pfalzgraf	she/her	alexispf
Teaching Assistant	Brian Tao	he/him	briantao
Teaching Assistant	Leanne Tong	she/her	leannet
Teaching Assistant	Jake Gustin	he/him	gustinj
Course Assistant	Gwen Liu	she/they	gwenl
Course Assistant	Paula Lopez Burgos	she/her	paulalb
Course Assistant	Daniel Wang	he/him	dxwang
Course Assistant	Kit Chung Yan	he/him	kyan
Course Assistant	Michelle Sun		mlxs
Course Assistant	Sebastian Wu	he/him	sebwu
Course Assistant	Shahnawaz Fakir	he/him	sfakir
Course Assistant	Jacob Stein	he/him	jmstein
Course Assistant	Mofolaoluwarera (Fola) Oladipo	he/him	foladipo

Midterms: Two 75 minute in-class midterms that will be held on **October 5, 2023(Midterm I)** and **November 7, 2023(Midterm II)**.

Your first written assignment is to read, sign, date (see page 16), and submit a copy of this syllabus to gradescope. See PS0 on Gradescope and Piazza for details.

Weekly Schedules: Lectures, Discussions, Office Hours and Piazza coverage

The following documents the weekly schedule for this semester. This schedule is also available as a google calender that you can subscribe to here: CS210 Google Calendar (you will need to be signed in with your BU google id to access this link).



Lectures and Discussions

As per the student link, this course is composed of two lectures per week, as well as discussions. It is critical you attend the discussion you are registered for, as we will be taking attendance based on the official rosters.

Office Hours and Piazza Coverage

In addition to the lectures and discussion, there are approximately 58 hours of support per week: 25 hours of office hours and 37 hours of Piazza staff coverage.

Office Hours: General office hours will be in: **Location:** See Piazza. The instructors' office hours are: **Location:** See Piazza **Times:** Tuesday 11:30AM-12:30PM and Thursday 11:30AM-1:00PM. If you need to arrange a time outside of these hours to speak with them, post privately on Piazza.

piazza: Staff piazza coverage, outside of the times noted, will be best effort.

The detailed weekly calendar at the end of this syllabus includes assignment due dates and midterm dates. Please plan your time appropriately.

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 underpin all computer systems. As part of this journey, we will teach you the attendant skills that you need to explore and solidify these concepts. These skills will distinguish you as professionals that know how things work and who know 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 underlie 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 reinforce 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 follow 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 somewhat 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, we 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 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 seems 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 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 computer systems is full 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 and be sure to plan accordingly.

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 a User Process is, 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 model, 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 material 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://cs-210-fall-2023.github.io/UndertheCovers/textbook

lecture notes: https://cs-210-fall-2023.github.io/UndertheCovers/lecturenotes

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://cs-210-fall-2023.github.io/CS210-Website/
2. Piazza site: https://piazza.com/bu/fall2023/cs210
   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 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/572191
   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. UNIX Development Environment: https://rhods-dashboard-redhat-ods-applications.apps.awscas210.zkf1.p1.openshiftapps.com/notebookController/spawner
   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 Piazza and the UCSLS textbook for information about how to use the environment.
5. GitHub Classroom: https://classroom.github.com/classrooms/140082261-cs-210-fall-2023-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 a github.com account, you will need to create one. Additionally, you will need to do a one-time registration to the github classroom. Doing this setup will be part the first assignment. Please post to Piazza if you have any questions or difficulties.

Piazza

We have purchased and set up a course Piazza site which can be accessed here:
https://piazza.com/bu/fall2023/cs210.

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 supersede 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 possible changes
- 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 where 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 a question on an assignment means or what is expected of you, please post.

Followups: If something was discussed in lectures, discussions, or mentioned in a reading and you would like to know more, please post.

Guidance: If there is a topic about computer systems that you would like guidance on 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 changing the visibility as needed. Please keep the following guidelines in mind.

- Be polite and considerate.
- Always search to see if the topic of your question 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 question that is a repeat please help your fellow student by directing them to the prior post.
- 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.

Computer Requirements

This course requires that you have a laptop on which you can access the internet using the Google Chrome web browser. If your computer breaks, then Information Services & Technology can help you with a temporary computer while you arrange a replacement.

COVID-19/Health Procedures

To promote a safe learning environment, students must adhere to the current University policies. At the time of writing the current policies can be found here:
https://www.bu.edu/shs/covid-19/.

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 week 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 components throughout the semester. All bonuses will be due on the last day of classes. Separate 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 outlined 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.

To be eligible for a particular bonus you must have submitted the corresponding assignment prior to the late date.

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:

October 5, 2023 and November 7, 2023.

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 (see student link). 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 assignment, PS0, is an introductory assignment to make sure that you are setup on all the course infrastructure and you will need to complete both parts **prior** to the first discussion. Please see posted information on Piazza.

The remaining six assignments, PS1 to PS6, form the core assignments for the class. The due dates can be found in the detailed calendar at the end of this document. Both parts A and B of a problem set will be released well in advance. Part A, the written component, will generally be due first, while the programming portion, Part B will be generally due one week later. All assignments are due at 11:59PM. Please consult the detailed weekly syllabus for the schedule. If there are any changes to the assignment schedule, we will post updates on Piazza.

We have provided a detailed schedule, stating when office hours and staff piazza coverage are, plan your time appropriately.

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 it 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 latter you will need to scan, or take pictures, of your paper copy to upload to gradescope.

Programming

The repository for the the programming 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. You will need to submit a copy of your main git repository on github classroom to this submission site. 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 timestamped 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. We will evaluate the messages, number, timing and contents of your commits. 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 teaching staff 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. As per the schedule, please do not expect staff responses outside of the Piazza staff coverage hours.

Lectures

The topics of the lectures build on each other. You will find it very difficult with later topics if you 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 in-discussion quizzes and monitor attendance.

Teaching Staff 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 and will be taken using TopHat.**

In addition to the discussion, the Teaching Fellows and Course Assistants will hold office hours. See the second 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 (tentative)

The following is a detailed weekly break down for the semester. The staff will provide updates and additional material on piazza as needed. **The following calendar contains clickable links that you can use to access the lecture slides and readings. In cases where no link exists, materials will be provided on Piazza**. The lectures for the last portion of course, which covers 'C' will be provided, as a pdfs on piazza. This material includes coverage of syntax and topics in the **CPAMA** book. See the lecture pdfs for details.

Date	Activity / Topics	Readings	Assignments
Tue 09/05/23	LEC1: Course overview and introduction	None	**PS0: OUT**
Thu 09/07/23	LEC2: UNIX Introduction and Preliminaries	ch1, ch2, ch3, 17.1, 4.1-4.6	
Fri 09/08/23			**PS0: DUE**
Mon 09/11/23	DIS1: Unix 1: Files and Directories, Redirection and		**PS1: OUT**
Tue 09/12/23	LEC3: Programming, Files and Directories	4.7-4.11	
Thu 09/14/23	LEC4: I/O, Process Control and Credentials	ch5	
Fri 09/15/23			**PS1A: DUE**
Mon 09/18/23	DIS2: Unix 2: Git, Text Editor, and Shell scripting		
Tue 09/19/23	LEC5: Editors, Make and Terminal IDEs		
Thu 09/21/23	LEC6: Version Control and GIT: The Basics		**PS2: OUT**
Fri 09/22/23			**PS1B: DUE**
Mon 09/25/23	DIS3: GDB 1: Intro: Starting and Basics		
Tue 09/26/23	LEC7: Assembly Programming Introduction	ch12, ch13	
Wed 09/27/23			**PS2A: DUE**
Thu 09/28/23	LEC8: Writing some simple assembly programs	ch13, ch14, ch15	
Mon 10/02/23	DIS4: GDB 2: Learn how to create executables and use GDB to explore CPU instructions		
Tue 10/03/23	LEC9: Assembly: Operations and Data Types	ch17	

Thu 10/05/23	Midterm I: In Class		
Mon 10/09/23	Indigenous People's Day, Classes Suspended		
Tue 10/10/23	Last Day to Drop Standard Courses (without a "W" grade) Substitute Monday Schedule of Classes DIS5: Piazza		**PS2B: DUE** **PS3: OUT**
Thu 10/12/23	LEC10: Assembly : Program Anatomy I		
Fri 10/13/23			**PS3A: DUE**
Mon 10/16/23	DIS6: Assembly 2: Conditionals, Loops and Multiple Source files		
Tue 10/17/23	LEC11: Program Anatomy II : Functions		
Thu 10/19/23	LEC12: Program Anatomy III : Code as Data		
Fri 10/20/23			**PS3B: DUE**
Mon 10/23/23	DIS7: Assembly 3: ASCII Strings, Heterogeneous Data Structure Example and Writing to standard output		**PS4: OUT**
Tue 10/24/23	LEC13: Program Anatomy IV: The Tree of Bytes and Data Structures		
Thu 10/26/23	LEC14: Assembly using the OS		
Mon 10/30/23	DIS8		
Tue 10/31/23	LEC15: Virtual Memory		
Wed 11/01/23			**PS4B: DUE**
Thu 11/02/23	LEC16: Processor Caches		**PS5: OUT**
Mon 11/06/23	DIS9: Practice Exam Review		
Tue 11/07/23	Midterm II: In Class		
Thu 11/09/23	LEC17: Fundamentals of C programming	CPAMA 1-6, 10	
Mon 11/13/23	Last Day to Drop Standard Courses (with a "W" grade) DIS10:C 1: C and Assembly		

Tue 11/14/23	LEC18: Functions & Arrays	CPAMA 8-9	
Wed 11/15/23			**PS5A: DUE**
Thu 11/16/23	LEC19: C types	CPAMA 7	**PS6: OUT**
Mon 11/20/23	DIS11: C Linked List in C		
Tue 11/21/23	LEC20: C pointers	CPAMA 11-13	
Thu 11/23/23	Thanksgiving Recess, Classes Suspended No Lecture		
Mon 11/27/23	DIS12:C Tic-Tac-Toe: Arrays and Recursion		**PS6A: DUE**
Tue 11/28/23	LEC21: C structures, unions, enums	CPAMA 16	
Thu 11/30/23	LEC22: Bitwise operations	CPAMA 20	
Mon 12/04/23	DIS13		
Tue 12/05/23	LEC23: Dynamic memory allocation	CPAMA 17.1-5	
Wed 12/06/23			**PS6B: DUE**
Thu 12/07/23	LEC24: Advanced uses of pointers	CPAMA 17.6-7	
Mon 12/11/23	DIS14: Practice Final Exam Review		
Tue 12/12/23	LEC25: Systems Research DIS14:Practice Final Exam Review - Attendance Optional		Last day for bonus sub-missions