



CS107, Lecture 1

Welcome to CS107!

readings:

[Course Syllabus](#)

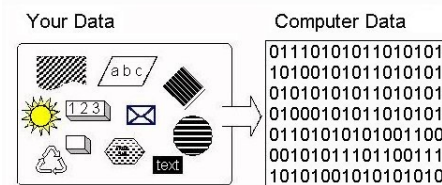
Bryant & O'Hallaron, Ch. 1 (skim)

[Honor Code and Collaboration Page](#)

CS107 Asks How and Why?

CS106B teaches you to solve problems using high-level programming languages.
CS107 digs deeper into **how** and **why** programs truly work.

- How is data (e.g., an **int**, a string, a **struct**) represented on hardware?
- How does a computer execute a program?
- How does an executable map into memory, onto hardware?
- How does the heap work, and how is it implemented?
- Why is my code doing one thing when I expect it to do something else?



Studying at this level of detail reveals why systems work the way they do and helps us become better software engineers.

CS107 Learning Goals

What do we hope you get out of the course? We'd love for students to:
attain **fluency** with

- ❤️ pointers and memory ❤️, and how to use them effectively
- an executable's address space and runtime behavior

develop **competency** with

- the translation between C code and assembly
- the implementation of programs that respect the limits of computer arithmetic
- the ability to identify bottlenecks and improve runtime performance
- the ability to navigate your own Unix development environment
- several ethical frameworks to consider when designing and implementing software

gain **exposure** to

- the basics of computer architecture
- compilers and assemblers and how they work

CS107's Topics and Big Questions

1. **Bits and Bytes**: How can a computer represent **int** values? or **floats**?
2. **Characters and C Strings**: How can a computer represent and manipulate more complex data like text?
3. **Pointers, Stack and Heap Memory**: How can we effectively manage all forms of memory in our programs?
4. **Generics**: How can we leverage our understanding of memory and data representation to write code that works with **any** data type?
5. **Assembly**: How do compilers translate C programs to assembly, and how do they execute? And what does assembly code look like?
6. **Heap Allocators**: How do core memory allocation functions like **malloc** and **free** work? Are the built-in versions always good enough?

CS107 Teaching Team



Instructor: Jerry Cain (jerry@cs.stanford.edu):

- Chemistry undergrad MIT, began a chemistry Ph.D. here, switched to CS in 1994
- Taught CS107 for the first time in 1999, have taught it some 30 times since



Companion Class: CS107ACE

CS107ACE is an extra, single-unit discussion section providing additional support, practice, and instruction.

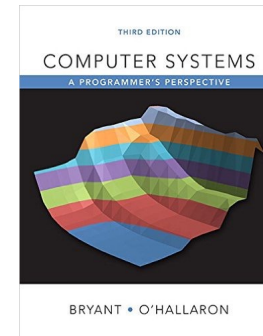
- Class meets on Tuesdays & Thursdays from 1:30 – 2:22pm in Littlefield 104.
- Admission is by [application](#), and everyone is welcome to apply. In past quarters, most students who've applied have been accepted.



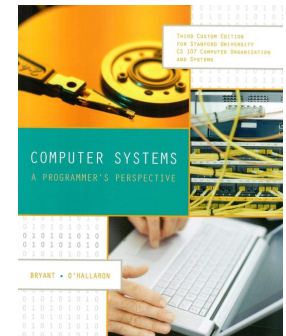
Isabel Berny

Textbooks

- *Computer Systems: A Programmer's Perspective* by Bryant & O'Hallaron, **3rd Edition**
 - **3rd edition matters** – important updates to content
 - Stanford Library has scanned **all** readings for CS107 under "fair use" (private study, scholarship, research). [**Canvas -> Files**].
 - If you want more, you can purchase a full copy.
- Any C programming reference of your choice
 - *The C Programming Language* by Kernighan and Ritchie (free link on course website Resources page)
 - Other C programming books, websites, or reference sheets



Full textbook



CS107 full chapters



canvas CS107-specific readings

The textbook and our C programming references are solid resources, and especially helpful post-midterm!

Course Structure

- **Lectures:** understand concepts, see demos, take weekly quizzes
- **Labs:** learn tools, study code, discuss with peers  **Great preamble to homework!**
- **Assignments:** build programming skills, synthesize lecture/lab content

	Monday	Wednesday	Friday
Week N		Lecture: part A	Lecture: part B
Week N + 1	Lecture: part C		

CS107 labs will meet at various times on Thursdays, and you'll use that time to more fully digest and exercise the prior week's material.

- **assign0:** goes out Friday and falls due on Monday, October 5th (requires today's, Friday's, and some of Monday's material)

CS107 Grading

*****	20%	Assignments
*****	20%	Lecture Quizzes
*	5%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

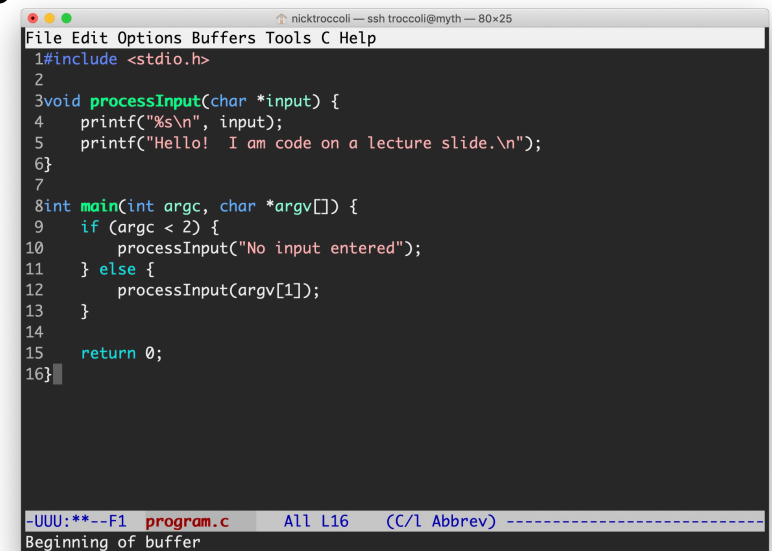


CS107 Grading: Assignments

****	20%	Assignments
****	20%	Lecture Quizzes
*	05%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

CS107 Assignments

- 7 programming assignments completed individually using **Unix command line tools**
 - Free software, already installed on **myth** machines / available on course website
 - We supply starter projects for each assignment
- Graded on **functionality** and **style**
 - Functionality graded using **automated tools**, given as point score, with minimal CA review
 - Style graded via code review (with **occasional automated tests**) given as bucket score
 - Grades posted to website and shared via email



```
File Edit Options Buffers Tools C Help
1#include <stdio.h>
2
3void processInput(char *input) {
4    printf("%s\n", input);
5    printf("Hello! I am code on a lecture slide.\n");
6}
7
8int main(int argc, char *argv[]) {
9    if (argc < 2) {
10        processInput("No input entered");
11    } else {
12        processInput(argv[1]);
13    }
14
15    return 0;
16}

-UUU:***-F1 program.c All L16 (C/l Abbrev) -----
Beginning of buffer
```

CS107 Late Assignment Policy

All programming assignments are due a minute before midnight, Stanford time, on whatever day they fall due.

- If you need to submit an assignment after the deadline, you still can. Doing so, however, imposes a cap on the maximum number of points you can get, depending on how late the submission.
 - Submit an assignment before the published deadline? You can get **100%**. Seems right.
 - Submit after the deadline, but within 24 hours? Functionality score is capped at 95%.
 - What does **capped** mean here? It means that all scores between 96% and 100% are demoted to 95%, but all other scores are left alone.
 - Submit an assignment between 24 and 48 hours post-deadline? Score capped at 90%.
 - Submit an assignment between 48 and 72 hours post-deadline? Score capped at 85%.
 - Unless you've made prior arrangements with us, no work is accepted after 72 hours.

CS107 Assignment Resubmission Policy

For all assignments except assign6, I will allow anyone who pulled less than 85% of the functionality points to resubmit to get to up to 85%.

You must, however, have received at least 25% of the functionality points in the first place. Otherwise, the maximum you can get is three times your original score.

- We only rerun the automated tests.
 - We don't review your code again or reread responses to short answer questions.
- Once grade reports for an assignment are shared with the class, you have seven calendar days to resubmit.
- Email me when you'd like to resubmit and I'll let you know how.

CS107 Grading: Lecture Quizzes

****	20%	Assignments
****	20%	Lecture Quizzes
*	05%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

CS107 Lecture Quizzes

- Beginning in Week 2, each Wednesday lecture will begin with a short quiz designed to exercise your understanding of the most recently taught material and the most recent assignment to have fallen due.
 - Quizzes are **seven minutes**, on pencil and paper, and are closed note, closed book, closed electronic device.
 - Each of your two lowest scores counts half as much as each of the others.
- These quizzes are low-stakes opportunities to verify your understanding.
 - Anyone who completed the most recent homework assignment and is up to date on lecture should find quizzes to be delightfully straightforward.

**All quizzes begin promptly at the beginning of lecture and end precisely seven minutes later, and the quiz ends when it ends.
If you miss the quiz, you take a zero.**



CS107 Grading: Labs

****	20%	Assignments
****	20%	Lecture Quizzes
*	5%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

CS107 Labs

- Weekly, in-person labs led by CAs, starting **next week**, offered on Wednesday evenings, Thursdays, and Friday mornings.
- Hands-on practice with lecture material and course concepts.
- Graded on attendance + participation. You get points for literally showing up **on time**, working through some exercises with everyone else, and staying for the duration.
- CGOE students complete their lab work (and take their quizzes and exams) remotely. More info in the [CGOE Handout](#).
- Lab preference submissions are already open, but you take your time finalizing your schedule before submitting. You may submit your preferences anytime until **Sunday 09/27 at 5PM PST**.

CS107 Grading: Exams

****	20%	Assignments
****	20%	Lecture Quizzes
*	05%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

CS107 Exams

- **Midterm exam:** Thursday, October 29th, 7:00 - 9:00pm outside of class
Thursday, October 29th, 3:30 - 5:30pm if first time is bad
 - Email me by Friday, October 23rd if you have an academic or extracurricular conflict with **both** times and absolutely cannot make either.
- **Final exam:** Tuesday, December 8th, 8:30 - 11:30am
 - If **and only if** you hold a conflict with this time because of a competing final exam—that is, you're taking two MWF 2:30pm classes and your other class also has a final—can you take the final on the same day, on December 8th, from 12:15 – 3:15pm instead.
- Both exams are **closed book, closed notes, and closed electronic device**. You'll be provided with a reference sheet with common function prototypes and other pertinent information not easily memorized.
- Both the midterm and final are administered as pencil-and-paper exams.

CS107 Grading

****	20%	Assignments
****	20%	Lecture Quizzes
*	5%	Lab Participation
****	20%	Midterm Exam
*****	35%	Final Exam

Read our full course policies document:

<https://cs107.stanford.edu/syllabus.html>

Getting Help

- Post on the **Ed Discussion Forum**
 - Online discussion forum for students, post questions, answer other student questions
 - Best for course material discussions, policy questions, debugging questions or general assignment concerns. **Never post assignment code to Ed.**
- Visit **Office Hours** in CoDa Basement
 - Chat about course topics or just hang out
 - Sign up in queues for 1:1 CA help, schedule to soon be posted on course website
 - Best for **group work**, **debugging questions** (with CAs, of course) or **longer discussions about course material**
- **Email** the Course Staff
 - cs107-win2627-staff@lists.stanford.edu or individual staff members
 - Best for **private concerns** (e.g., grading questions, academic accommodations, etc.)
- OAE letters should be sent to jerry@cs.stanford.edu as soon as possible.

Stanford Honor Code

- The [Honor Code](#) is an undertaking of the Stanford academic community, individually and collectively. Its purpose is to uphold a culture of academic honesty.
 - Students will support this culture of academic honesty by neither giving nor accepting unpermitted academic aid in any work that serves as a component of grading or evaluation, including assignments, examinations, and research.
 - Instructors will support this culture of academic honesty by providing clear guidance, both in their course syllabi and in response to student questions, on what constitutes permitted and unpermitted aid. Instructors will also not take unusual or unreasonable precautions to prevent academic dishonesty.
 - Students and instructors will also cultivate an environment conducive to academic integrity. While instructors alone set academic requirements, the Honor Code is a community undertaking that requires students and instructors to work together to ensure conditions that support academic integrity.

(drawn from <https://communitystandards.stanford.edu/policies-guidance/honor-code>)

It is your responsibility to read and become familiar with the honor code guidelines on the CS107 course website. Please read them and come speak with us if you have questions.

<https://cs107.stanford.edu/collaboration>

Stanford Honor Code and CS107

- Be fully transparent about how you get your work done
 - Indicate any assistance you received on homework (books, websites, friends, etc.).
 - Don't look at another student's solution code or answers and never share solutions with others or post them to the web.
 - Never use LLMs to generate assignment code or answers unless explicitly permitted.
 - This replaces the intellectual work the assignment is designed to assess and therefore constitutes a serious violation of the Honor Code.
- Assignment submissions are regularly examined using software tools.
- If you need help, contact us and we'll help you make good choices.
 - If you feel you've made a mistake, you may retract any assignment submission at any time before the final exam by simply emailing me.

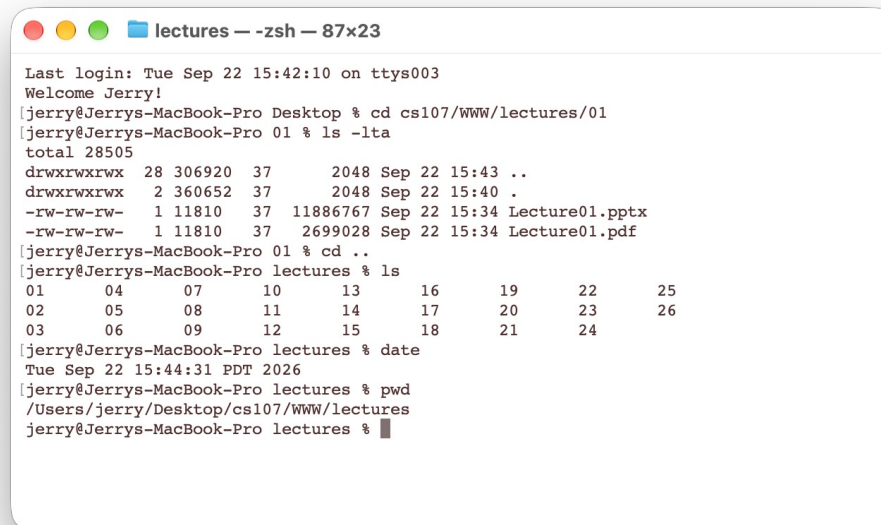
<https://cs107.stanford.edu/collaboration>

AI, LLMs, and CS107

- **Don't outsource your CS education to AI.** You can't reliably evaluate AI-generated code until you understand the computer science underneath it.
 - Getting working code isn't the same as learning how to program.
 - The struggle (designing, debugging, tracing, repairing) is where the learning happens.
 - LLMs can be **confidently wrong**, so true expertise still matters.
- **Even experienced programmers misjudge how much AI is helping them.**
 - Some companies ([Meta](#), [Ford](#), [Klarna](#), [Commonwealth Bank of Australia](#)) have reversed AI-driven staffing decisions after AI-driven automation fell short of expectations.
 - Others ([Snowflake](#), [LinkedIn and others](#)) are increasing intern and new-grad hiring rather than assuming AI can replace junior engineers.
- **The goal isn't to avoid AI forever.** That's akin to resisting Google in 1997.
 - Develop **enough of domain expertise** to eventually—post CS107, of course—use AI well.

What is Unix?

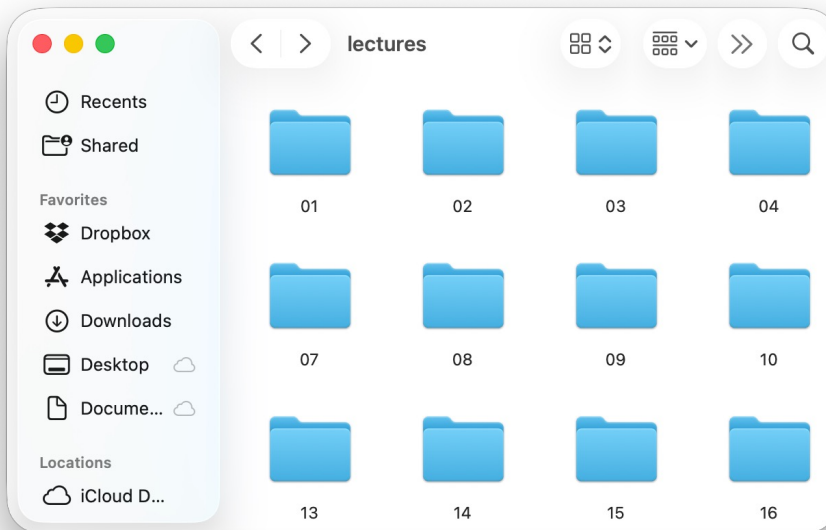
- **Unix:** provides a set of standards and many tools used to write software.
 - **macOS** and **Linux** are operating systems built on top of Unix
- You navigate a Unix system using the **command line** (aka "**the terminal**")
- All Unix systems work using the same tools and commands



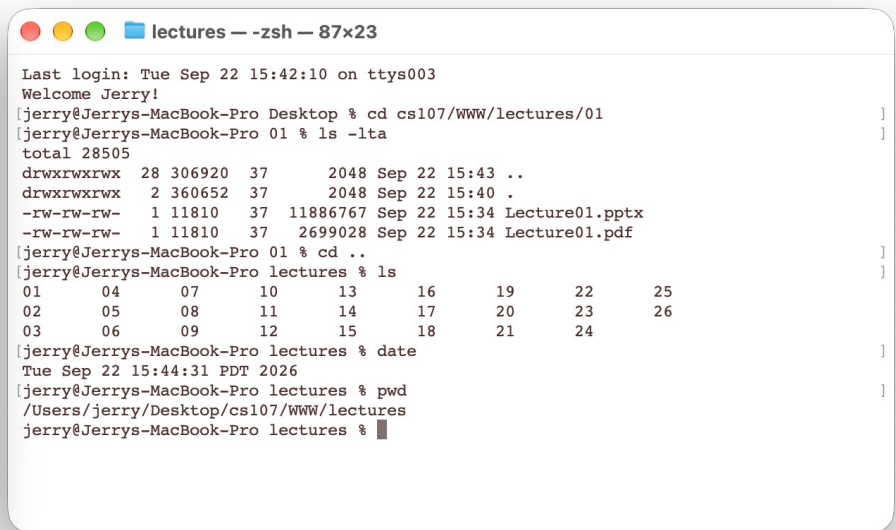
```
lectures --zsh -- 87x23
Last login: Tue Sep 22 15:42:10 on ttys003
Welcome Jerry!
[jerry@Jerrys-MacBook-Pro Desktop % cd cs107/WWW/lectures/01
[jerry@Jerrys-MacBook-Pro 01 % ls -lta
total 28505
drwxrwxrwx  28 306920  37      2048 Sep 22 15:43 ..
drwxrwxrwx   2 360652  37      2048 Sep 22 15:40 .
-rw-rw-rw-   1 11810   37 11886767 Sep 22 15:34 Lecture01.pptx
-rw-rw-rw-   1 11810   37 2699028 Sep 22 15:34 Lecture01.pdf
[jerry@Jerrys-MacBook-Pro 01 % cd ..
[jerry@Jerrys-MacBook-Pro lectures % ls
01      04      07      10      13      16      19      22      25
02      05      08      11      14      17      20      23      26
03      06      09      12      15      18      21      24
[jerry@Jerrys-MacBook-Pro lectures % date
Tue Sep 22 15:44:31 PDT 2026
[jerry@Jerrys-MacBook-Pro lectures % pwd
/Users/jerry/Desktop/cs107/WWW/lectures
[jerry@Jerrys-MacBook-Pro lectures %
```

What is the Command Line?

The **command line** is a text-based interface to navigate a computer's file system and launch executables. It's a popular, lightweight alternative to the more traditional graphical user interfaces you've used before.



Graphical User Interface

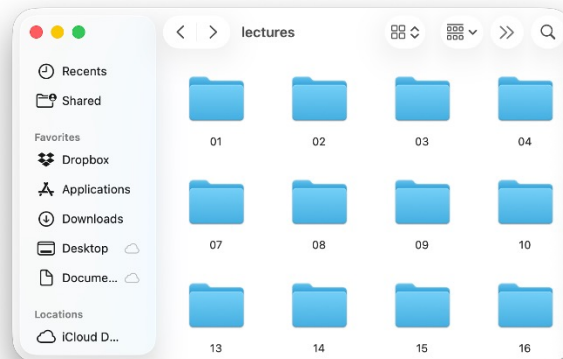


Command line interface

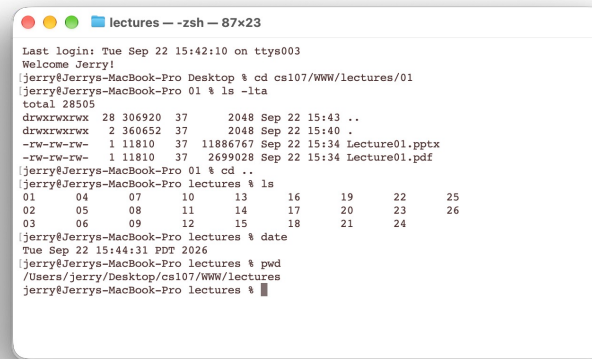
CLI vs. GUI

Just like a GUI file explorer, a terminal:

- shows you a **specific place** on your computer at any given time
- lets you descend **into folders** and **ascend out of them**
- lets you **create new** files and directories and **edit** existing ones
- lets you **execute programs**



Graphical User Interface



Command-line interface

Why Unix?

- You can navigate virtually any device using the same tools and commands:
 - laptops and desktops
 - embedded devices (Raspberry Pi, etc.)
 - mobile Devices (Android, etc.)
 - microwaves, dishwashers, air conditioners, cameras
- Used frequently by software engineers for:
 - **web development**: running servers and web tools on servers
 - **machine learning**: processing data, training models
 - **systems**: implementing operating systems, networks, compilers, and drivers
 - **mobile development**: running tools, managing libraries
- We'll use Unix and the command line to implement and test C programs.

Unix Commands To Try

- **cd**: change directories
- **ls**: list directory contents
- **mkdir**: create a new directory
- **emacs**: open text editor
- **rm**: irreversibly delete file or folder
- **man**: read documentation for standard C functions

See the course website
for more commands and
a complete reference.

Demo: Using Unix and the Command Line



Get up and running with our guide:

<http://cs107.stanford.edu/getting-started.html>

Learning Unix and the Command Line

- Using Unix and the command line can be very intimidating at first.
 - How do I know what to type?
 - It looks retro! It **is** retro!



- You're learning a new language.
 - You should repeatedly look things up ([resources](#) on course website)
 - Spend as much time as possible (during labs and assignments) building muscle memory with the tools



Question Break

Get up and running with our guide:

<http://cs107.stanford.edu/getting-started.html>

assign0

assign0 (Intro to Unix and C) goes out on Friday and falls due ten days later, on **Monday, October 5th at 11:59PM PDT**.

There are **5** parts to the assignment, which is meant to get you comfortable using the command line, and editing/compiling/running C programs:

- surveying the course website to learn a few different Unix commands
- **clone** the **assign0** starter project
- **answer** several questions in **readme.txt**
- **compile** a provided C program and **modify** it
- **submit** the assignment

Lecture Recap

CS107 is a programming class in C that teaches you about programming languages and software and what goes on under the hood.

- We'll use Unix and its tools to implement, debug and test our programs.
- Please visit the course website, <https://cs107.stanford.edu>, where you can read the lecture schedule, the syllabus, the information we provide about the Honor Code in CS107, and much, much more.

I'm looking forward to an awesome quarter!

