Calendar

Below is a preview of the week-by-week plan for the quarter. There may be adjustments and rearrangements as we go.

- In the readings listed below, B&O is Computer Systems (Bryant and O'Hallaron), K&R is The C Programming Language (Kernighan and Ritchie) **accessible here (https://openlibrary.org/books/OL2030445M/The_C_Programming_Language)** (requires free Open Library account to borrow), and Essential C is a PDF available at **http://cslibrary.stanford.edu/101 (http://cslibrary.stanford.edu/101)**. B&O scanned PDFs for the listed readings are available on Canvas **under the "Files" tab (https://canvas.stanford.edu/courses/197439/files)**.
- Lecture videos are posted on Canvas **under the "Course Videos" tab (https://canvas.stanford.edu/courses/197439/external_tools/3367)**.

Lecture Code

Any code examples worked in class will be posted after lecture into `/afs/ir/class/cs107/lecture-code/lect[N]` where you replace `[N]` with the lecture number. You can make a copy to compile or modify by doing the following, which will make a folder in the current location called `lect[N]` that is a copy of the `lect[N]` code.

```
cp -r /afs/ir/class/cs107/lecture-code/lect[N] lect[N]
```

You can also view lecture code from your web browser by clicking **here (lecture-code)**.

Topics	Readings	Assignments
Week 1		
Lecture 1 (Mon 9/23): Welcome to CS107! We'll go through course logistics and discuss the Honor Code.	**Honor Code (/class/archive/cs/cs107/cs107.1252/collaboration)** B&O Ch 1 - skim this chapter for a quick overview of what is meant by systems, and for a preview of topics to come.	
No labs this week.		
Lecture 2 (Wed 9/25): Tour of Unix, Basic C Programs, Integers and Data Representation **Topic 1:** *How can a computer represent integer numbers?* After a whirlwind tour of Unix and the basic structure of a C program, we'll learn more about the representation of the integer types: `char`, `short`, `int`, and `long`, in both unsigned and two's complement signed. We'll also discuss integer arithmetic, overflow, truncation and sign extension, and how mixed signed and unsigned comparison operations work.	B&O Ch 2.2-2.3 - skim the formal proofs, but it's important to take away a solid working knowledge of two's complement and behaviors of integer operations.	**Out**: assign0
Lecture 3 (Fri 9/27): Integers and Data Representation, Take II We'll continue our discussion of data representations and work through additional examples to solidify your understanding of the most basic data abstractions needed to code	B&O Ch. 2.2-2.3	
Week 2		
Lecture 4 (Mon 9/30): Bits and Bitwise Operators We'll dive further into bits and bytes and how to manipulate them using bitwise operators.	B&O Ch 2.1	
Lecture 5 (Wed 10/2): Bitwise Operations, Chars and C-Strings **Topic 2:** *How can a computer represent and manipulate more complex data like text?* We'll use most of the time to cover bitwise operations. Time permimtting, we'll start to explore how strings and characters are manipulated in C using the `char` and `char *` types, discuss null termination, and become familiar with the `string.h` functions.	K&R (1.9, 5.5, Appendix B3) or Essential C section 3 for C-strings and `string.h` library functions. C-strings are primitive compared to Java/C++ strings, so take note of the manual efforts required for correct use and pitfalls to avoid.	**In**: assign0 **Out**: assign1
Lab 1: Bits and `ints`	Be sure to check out our **guide to gdb (/class/archive/cs/cs107/cs107.1252/resources/gdb)**.	
Lecture 6 (Fri 10/4): Chars and C-Strings, Take II We'll continue working with the `char` and C strings ahead of our initial work with general arrays and pointers.	K&R (1.9, 5.5, Appendix B3) or Essential C section 3 for C-strings and `string.h` library functions.	
Week 3		
Lecture 7 (Mon 10/7): More C-Strings Now it's time to start digging into the use of `*` and `&`, pointer operations/arithmetic, and memory diagrams. In lecture, we will trace through code, draw lots of pictures, and poke around in gdb.	K&R Ch 1.6, 5.5 or Essential C section 3 on the mechanics of pointers and arrays. Pay special attention to the relationship between arrays and pointers and how pointers/arrays are passed as parameters.	
Lecture 8 (Wed 10/9): Even More C-Strings Advanced work with pointers, double pointers, and arrays of C strings.	Finish K&R Ch 1.6, 5.5 or Essential C section 3	**In**: assign1 **Out**: assign2

Topics	Readings	Assignments
Lab 2: C-Strings	Be sure to check out our **guide to Valgrind (/class/archive/cs/cs107/cs107.1252/resources/valgrind)**.	
Lecture 9 (Fri 10/11): Arrays and Pointers **Topic 3:** *How can we effectively manage all types of memory in our programs?* We'll answer questions like: how are arrays and pointers the same? How are they different? How is an array/pointer passed/returned in a function call? After this lecture, you'll understand how arrays and pointers allow two syntaxes for accessing sequential memory locations, but the underlying reality of the memory is the same.	K&R 5.2-5.5 or Essential C section 6 on advanced pointers	
Week 4		
Lecture 10 (Mon 10/14): Stack and Heap We'll learn about stack allocation, stack frames, and parameter passing. Then, we'll introduce dynamic allocation on the heap (`malloc` / `realloc` / `free`), heap contractual guarantees and undefined behavior.	K&R 5.6-5.9 or Essential C section 6 on the heap. The key concept is comparing and contrasting stack and heap allocation.	
Lecture 11 (Wed 10/16): More Stack and Heap We'll spend this lecture working through a collection of examples that necessarily rely on nontrivial dynamic memory allocation, thereby requiring we understand `malloc` , `realloc` , and `free` very, very well.	still working through K&R 5.6-5.9 or Essential C section 6 on the heap.	**In**: assign2 **Out**: assign3
Lab 3: Arrays/Pointers		
Lecture 12 (Fri 10/18): `void *` , Generics **Topic 4:** *How can we use our knowledge of memory and data representation to write code that works with any data type?* We'll continue comparing and contrasting stack and heap allocation. Then, we'll move on to untyped `void *` pointers and motivate C generics. We'll also discuss vulnerability disclosure and partiality.	still working through K&R 5.6-5.9 or Essential C section 6 on the heap.	
Week 5		
Lecture 13 (Mon 10/21): More Generics We'll talk about function pointers, which allow us to implement generic operations using client callbacks.	K&R 5.11, review man pages or your C reference to be introduced to generic functions in the C library (`qsort` , `lfind` , `bsearch`)	
Lecture 14 (Wed 10/23): Generic Data Structures We'll talk about function pointers, which allow us to implement generic operations using client callbacks.	finish K&R 5.11	**In**: assign3 **Out**: assign4
Lab 4: `void *` /Function Pointers		
Lecture 15 (Fri 10/25): Intro to x86-64, Data Movement **Topic 5:** *How does a computer interpret and execute C programs?* We'll introduce assembly/machine language and find out what's happening underneath the hood of the C compiler, including a discussion of the x86-64 instruction set architecture and its powerful `mov` instruction. Then, we'll talk about addressing modes, data layout, and access to variables of various types.	B&O 3.1-3.3 for background info on x86-64 assembly. Very carefully read B&O 3.4 on addressing modes and data transfer. The multitude of addressing modes is one of the things that puts the first "C" in CISC. Be sure to check out our **x86-64 guide (/class/archive/cs/cs107/cs107.1252/guide/x86-64.html)**.	
Week 6		
Lecture 16 (Mon 10/28): x86-64 ALU We'll talk about arithmetic and logical instructions.	B&O 3.5-3.6 Be sure to check out our **x86-64 guide (/class/archive/cs/cs107/cs107.1252/guide/x86-64.html)**.	
Lecture 17 (Wed 10/30): Advanced x86-64 ALU We'll talk about arithmetic and logical instructions.	B&O 3.5-3.6 We're still referring to **x86-64 guide (/class/archive/cs/cs107/cs107.1252/guide/x86-64.html)**. It's our best friend this week.	
Lab 5: Assembly x86-64 in all its glory	Be sure to check out our **x86-64 guide (/class/archive/cs/cs107/cs107.1252/guide/x86-64.html)**.	Midterm (Thu)
Lecture 18 (Fri 11/1): x86-64 Condition Codes and Control Flow We'll see how to implement C `if` / `else` and loops in assembly and discuss other uses of condition codes (`setx` / `cmov`).	B&O 3.6 Be sure to check out our **x86-64 guide (/class/archive/cs/cs107/cs107.1252/guide/x86-64.html)** for coverage of these new operations.	
Week 7		

Topics	Readings	Assignments
Lecture 19 (Mon 11/4): x86-64 Runtime Stack We'll talk about procedures and the runtime stack and introduce instructions for call/return, parameter passing, local stack storage, and register use.	B&O 3.7, 5.1-5.6	
Lecture 20 (Wed 11/6): x86-64 Runtime Stack, Take II We'll continue discussing the runtime stack to solidify concepts that will help inform your work on assign5.	B&O 3.7, 5.1-5.6	**In**: assign4 **Out**: assign5
Lab 6: Runtime Stack Fun explorations with the stack!		
Lecture 21 (Fri 11/8): Reverse Engineering, Privacy and Trust We'll discuss assign5's focus on reverse engineering executables as well as larger discussions of privacy and trust. We'll do lots of reverse engineering practice to get you up to speed on assign5!		
Week 8		
Lecture 22 (Mon 11/11): Managing the Heap **Topic 6:** *How do core memory-allocation operations like `malloc` and `free` work?* We'll see how the heap fits into the address space. We'll introduce design decisions for implementing `malloc` / `realloc` / `free` , as well as performance tradeoffs (throughput, utilization).	B&O Ch. 9.9 and 9.11 cover heap allocation implementation and memory misuses. There's lots of very useful detail in 9.9 for your heap allocator!	
Lecture 23 (Wed 11/13): Managing the Heap, Take II We'll have worked through two very simple allocator designs on Monday, and today we'll invest some time speaking about a third design, which is the one that's actually used in practice.	B&O Ch. 9.9 and 9.11	
No labs this week.		
Fri 11/15 - No Lecture		**In**: assign5, **Out**: assign6
Week 9		
Lecture 24 (Mon 11/18): Optimization We'll take a look at the compilation process and the various optimizations gcc does on our behalf. We'll also learn of some tools we can use to profile our own code so we know where to optimize ourselves. This could be helpful for final tuning of your heap allocator!	Optimally skim B&O Ch. 5	
Lecture 25 (Wed 11/20): Optimization, Caching We'll continue our discussion some optimization feastures not covered on Wednesday, and we'll discuss how to write cache-friendly code!		
Lab 7: Code and Memory Optimization Experiments in optimization and profiling	Be sure to check out our **CS107 guide to callgrind (/class/archive/cs/cs107/cs107.1252/resources/callgrind)**.	
Fri 11/22 - No Lecture		**In**: assign6a
Week 10		
Lecture 26 (Mon 12/2): Wrap-up and Q&A		
Wed 12/4 - No Lecture		**In**: assign6
No labs this week.		
Fri 12/6 - No Lecture		
Final Exam Week		
Final Exam: Mon 12/9, 3:30PM-6:30PM		