Final Overview:

- Final Exam Information
    - Time: Friday, December 15th 3:00 PM - 6:00 PM
    - Location: 5280 Boelter
    - Expected duration: 2.5 – 3 hours
- Question Types:
    - Multiple choice questions
    - Short essay questions (about concepts)
    - Writing code (python, C, or bash)
- How to study:
    - Review all labs and assignments (I won't be asking questions about specific assignments. You need to know and understand the concepts we discussed each week and be able to apply them to new problems)
    - material in the assignments as well as material covered in class can be on the exam
    - The final is open-notes, open-book (no electronic devices). Feel free to bring a copy of the slides, your lab logs, your study notes, or any books you have to the final.

This is a list of all the topics we covered each week. Go through each bullet point, review the corresponding slides, and make sure you understand the concepts.

Week 1:

- Basic Linux commands (you need to know and be comfortable with using these commands without needing any documentation)
    - ls (-l, -t), pwd, cd, mkdir, rmdir, echo (-e), cat, cp, mv, ln (-s), rm (-r), find (-type, -perm, -user, -atime, -mtime, -name, -maxdepth), chmod, kill, ps, whereis
- man pages
    - Sections and headings of a man page
    - how to find relevant info
- Unix permissions (chmod command), how to change permissions on a file
    - Symbolic
    - Numeric
- Piping and redirection
    - How to redirect stdin, stdout, stderr

Week 2:

- Regular expressions and wildcards
    - 4 concepts of regular expression and symbols associated with them
    - Difference between basic regular expressions and extended regular expressions
- Advanced Linux commands: find (-user, -perm, -name, -type, -atime, -mtime, -maxdepth), grep (difference between grep, egrep, fgrep), sed (substitution command, backreferencing), tr
- Bash scripting (a lot of material here):
    - different constructs: for loops, while loops, if statements
    - declaring and referencing variables, environment variables, etc.
    - built-in variable: #, ?, IFS, 1, 2, etc.
    - Types of quotes: Single vs. double quotes vs. back ticks
- Links (hard links vs. symbolic links, ln command)

Week 3:

- C compilation process (source code -> preprocessor -> compiler -> assembler -> linker)
- Makefiles (structure of rules, how does make work?) and the build process (configure, make, make install)
- Python scripting (defining classes, member functions, helper functions, variables, constructs, indentation, checking number of arguments, lists and list operations)
- Patch, diff
    - Applying patches: unified diff format, pnum option

      diff –u file1 file2 > patch_file
      --- /usr/local/bin/file1 (path to original)
      +++ /usr/local/bin/file2 (path to modified)
      @@ -l,s +l,s @@ hunk, l,s lines the change applies to in from/to files

      If you're in local, which num should you use? => patch –pnum < patch_file

Week 4:

- C programming: pointers, function pointers, qsort, file I/O, memory allocation, etc.
- GDB
    - How to run it: gdb <executable>, (gdb) run <args>
    - debugging commands (break, bt, watch, c/n/s, break file:line, bt (stack frames))
- How are stack frames built? How is the stack used?

Week 5:

- Protection mechanism (dual modes: user mode, kernel mode. Why does dual mode operation exist?)
- What are system calls?
- Disadvantage of system calls
- Library calls vs. system calls (getchar/putchar vs. read/write)

Week 6:

- Difference between multitasking and multithreading (advantages/disadvantages of each)
- Difference between a thread and a process
- Race conditions and thread synchronization (what are race conditions? What's the purpose of synchronization methods? Are there any synchronization methods we discussed in class?)
- pthreads interface (pthread_create, pthread_join)
- Need to be able to parallelize a program

Week 7:

- SSH protocol steps (Where is the symmetric/asymmetric encryption happening?)
- How does SSH guarantee confidentiality and authentication?
- Symmetric key cryptography (what is it? How does it work?)
- Asymmetric key cryptography (private/public keys)
- Digital signatures (how do they work? How do they guarantee data integrity?)
- In SSH, the key used for encryption is the receiver's public key. When creating a digital signature, the key used for encryption is the sender's private key. Why? What is a digital certificate?

Week 8:

- The process of linking (What is a linker's job?)
- Difference between static and dynamic linking (pros and cons of each)
- How to create static library vs. shared library
- Difference between dynamic linking and dynamic loading
- Dynamic Loading API (dlopen, dlsym, dlerror, dlclose)

Week 9:

- Git version control
    - Centralized vs. distributed VCS
    - Objects in git architecture (blob, tree, commit, tag)
    - 2 ways to create a version controlled project (git init, git clone)
    - Git commands: commit, checkout, branch
    - Why do you have to add then commit?
    - What's a branch? A pointer to a commit object
    - Default branch? Master. Keeps moving each time a commit is added
    - How to create a new branch. Why is it useful?

Not on final: EMACS, Man Pages