Announcements

- **Today: finish up discussion of multprocessing, launch into multithreading and concurrency.**
  - Assignment 3 due on Monday night.
  - Assignment 3 out on Monday night as well, due eight days later.
    - Will likely post the assignment handout and starter repos on Sunday night, so look for them online.
  - B & O Reading: Chapter 12, skipping section 12.2. (Chapter 12 is the fourth of the four chapters in the reader.)
  - I'll spend 10 or 15 minutes reviewing virtual memory, pages, translation lookaside buffers, schedulers, context switches, process control blocks, and ready and blocked PCB queues. I discuss all of these because:
    - I want you to understand how each process can operate as if it owns all of memory, even when there are hundreds of other processes that operate the same way.
    - I want you to understand how multiple processes can be executing at the same time, even though the machine you're on has a very small number of CPUs.
  - During the last hour or so, I'll teach you a small amount of threading using C and **`pthreads`**.
    - Textbook examples reference **`pthreads`**, which is why I want you to see at least a little bit of it in lecture before we abandon it and move over to C++.
    - Concepts taught in B & O Chapter 12 (which you are reading over the course of the next two weeks) are relevant in all languages, so don't go on thinking the reading won't be helpful.
    - I'll also illustrate the simplest of concurrency issues using **`pthreads`** so I can later argue why C++ threads, which is what

we'll spend most of our time learning, make things much easier (or at least less difficult, since anything having to do with threading is very challenging).

- Will start C++ threading once I get through three small **`pthreads`** examples.
- Be sure to inspect code samples in **`/usr/class/cs110/lecture-examples/spring-2017/threads-c`** to see and play with working versions of everything in the coming slide decks.