1. Operating systems (OSes)
    a. What's the objective of an operating system?
        i. Make the computer easy to use
            1. Provide services for program development
                a. For example, suspend and continue signals when using GDB
            2. Make starting a program simple
                a. Say, clicking an icon
            3. Provide security via password protection
            4. Handle error correction and detection transparently
                a. Correct errors in I/O and files
            5. Provide quick response time to the user
                a. Adjust resource assignments to prioritize application in foreground
        ii. Manage the computer's resources so they are used efficiently
            1. OS is just another program
                a. Relies on the processor and system like any other
                b. Relinquishes CPU to user processes, relies on CPU to give control back
                c. It is a more "privileged" program, however
            2. Kernel – the "core" of the OS
                a. Frequently used routines that stay in RAM all the time

2. History of operating systems
    a. Two dimensions determine how the OS runs things
        i. Interactivity of machine – batch versus interactive
            1. Interactivity is what we know today, interact directly with computer
        ii. Number of programs being worked on – uniprogrammed versus multiprogrammed
    b. Original OSes didn't do much
        i. Computer sat idle waiting for operator to enter programs, then ran the program
        ii. Uniprogrammed OS – only worked on one program at a time
    c. Batch idea introduced
        i. Was able to do this with some more RAM
        ii. Give all the programs to run at once to the operator, operator loads them in
        iii. Computer *monitor* responsible for loading program and running it
            1. Then it would print out results and load new program
        iv. Better than previous, but still not great
            1. Spent lots of time waiting for I/O due to difference in speeds
    d. Finally, multiprogrammed machines
        i. Takes even more RAM
        ii. Allows multiple programs available to run at once
            1. Not running *simultaneously*, but gives the appearance of that
        iii. Less waiting around
            1. Program stalls? Go to the next one
        iv. Need some more components for this
            1. Memory protection – prevent processes from overwriting another's data
            2. Memory management – not enough RAM to store entire program in memory at once
                a. Discuss this one next
            3. Scheduler – talk about this later if we have time