Operating System

Processes:

- instance of computer program
- abstraction for a running program
- single CPU can appear to execute multiple processes concurrently by having the processor switch among them
- OS performs this using contect switching
- OS keeps track all state information for a process (context)
  - values of PC, register file, main memory
- single processor can only execute code for single process
- when OS decided to transfer control to new process performs context switch
  - save contect of current process, restore contect of new process, pass control to new context
- system call: special function, passes control to OS and saves shell context, creates new context (hello) and gives it control
- scheduling: kernel decides to preempt the current process and restart a previously preempted process
  - handled by code in the kernel called scheduler
- sum: decision: scheduling, act: context switch

Threads:

- execution unit
- process can be made up of multiple threads that execute concurrently on process context with same code and global data
- requirement for concurrency in network servers make threads important
  - easier to share data between multiple threads than multiple processes
  - threads more efficient than processes
- multithreaing a method to make programs faster with multiple processors

Virtual Memory:

- abstraction provides each process the illusion that it has exclusive use of the main memory
  - heap: follows code and data areas (which are themselves fixed in size)
    - expands and contracts dynamically at runtime (malloc, free)
  - Shared libraies: near the middle
  - stack: at the top
    - used to implement function calls
    - expands and shrinks dynamically (grows when you call/shrinks when you return from a function)
  - kernel virtual memory

Sources:

- Computer Systems: A Programmers Perspective