Recitation 3: UNIX I

We cover UNIX in two parts. This part focuses on the filesystem and naming; the second part focuses more on processes and the shell. You should make sure you're comfortable with how UNIX resolves a file name (i.e., how it gets from the filename to the file data). Come to office hours if you have questions!

Background

- Before UNIX there was Multics. Multics was a relatively large, complex system. UNIX was a reaction to this, built by two people on a small machine. Used by the authors for their own research.
    - "Small machine" is *super* small compared to what we deal with today. Small files, 512-byte blocks, hundreds of megabytes of storage, hundreds of kilobytes of memory.
- Introduced the ideas of files, subdirectories, etc.

Naming

- Lots of names in UNIX: filenames, devices, i-numbers, physical addresses, addresses of blocks, user names, …
- How do we get from a filename (e.g., /home/user/file.txt) to the actual data of the file? Need directory entry, directory, i-list, i-node. Lots of layers of naming here.
    - What happens if the file is large?

Filesystems

- In UNIX, "everything is a file". So files are the basic abstraction used by UNIX.
- File descriptor = integer "handle" used by a process to refer to an open file. Identifies a kernel data structure that points to a position where the next byte will be read from/written to
- File = persistent storage, unstructured. Can create, open, read, write, delete files (also mount, link). Serial access, not random access.