

0 - std. input / files
 1 - std. output / files
 2 - std. errors

1 - curr. dir
 2 - parent dir
 / - the root

system calls - low level
 system libs - wrap

PIPE - only between child proc.

stored in a buffer in mem.

FIFO - can com. between programs

files on disk

symlinks - 'shortcuts'; reference to another file

- if deleted: points to nothing.

hard links - created only by the root

- exact replicas of the file

- if deleted: links still working having access to data

PROCESSES vs THREADS

→ pure area - does not get copied.
 impure - user + kernel + dead
 ↳ copies the main proc. after fork()

zombie proc. - proc. that has compl. exec. but is still in the proc. table.
 orphan: child alive, parent dead.

read: w8 for data or no writer connected

write: w8 for & space or no reader connected

* open: w8 for open in both ways (compar. op)

→ only creates a stack in memory

⊕ same memory.
 ⊕ IT shares data locally
 ⊕ IT has no security.

Threads ⇒ mutex lock

⇒ lock → waits for n threads

⇒ semaphore →

allow many th. can execute as in it simult.

Deadlocks - get sed. close a victim and stop the proc victim.

* go back to a check point

- Detect one: lock for cycles.

- prevent: keep the same order for opening/closing resources.

- Causes: mutual exclusion.

→ resources that wait for each other.

Semaphore — (V(s), C(s))

value of the sem & ↳ waiting list ↳ parameter

Heaplocking = associated a storage device to a particular loc. in the dir tree.

Caching

- **Direct cache**
 - + fast and simple
 - may lead to the cache thrashing (collision)
- **Associative cache**
 - place on each page on a free location. found by iter. through the cache
 - rather slow
- **Set associative cache**
 - hash table
 - the cache in org. in sets of pages
 - the set of a page is comp. on pg. address % cache set.
 - flexible; avoid collisions.

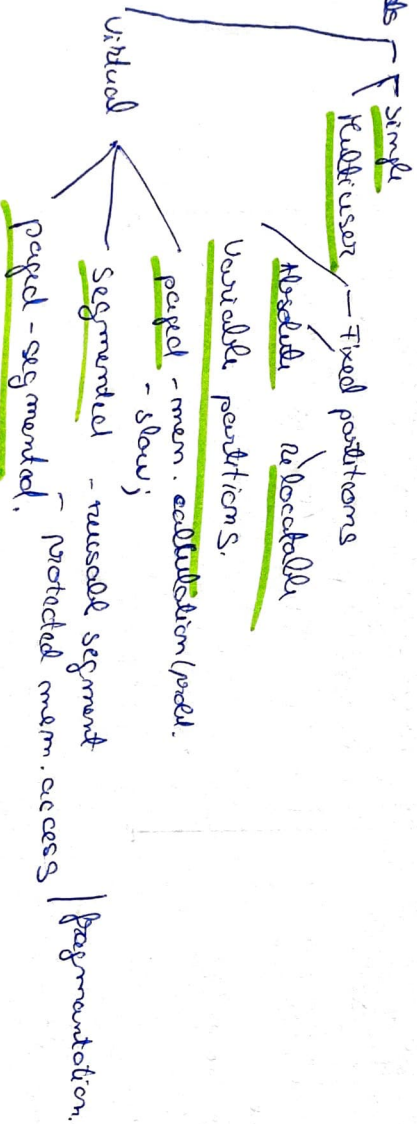
Process states:

- Hold**: - becomes a proc. but not executed
- hold back by the job planning and resource allocation.
- Ready**: loaded in memory but doesn't have a processor to execute it
- Run**: oscillates between run, ready
- Wait**: when doing I/O
- Sleep**: when memory full
- Finished**: resources are deallocated, process done.

Process scheduling

- **First come, first served**
- small proc. waits after the big ones.
- **Shortest job first**
- shortest jobs smother in
- **Priorities**
- by length, deadline
- **Round Robin**

Allocation methods



Choose a victim

FFFO; not recently used; least recently used (2 bits); most recently used