Scheduling

Linux-Kernel-Development[1]

Two flavours of multitasking OS:

- *cooperateive multitasking*
  - process does not stop until it itself voluntarily decides so
- *preemtive multitasking*
  - scheduler decides when a process ceases running and a new process begins running
  - the act of involuntarily suspending a running process is called preemption
  - time process runs before preempted is predetermined = *timeslice*
- Process classification (not mutually exclusive):
  - *I/O bound*
    - spends much of its time submitting and waiting on I/O
    - runs only shortly before it blocks waiting on I/O
  - *Processor bound*
    - majority of time spend executing code
- Process priority - Linux kernel has two separate priority ranges
  - *nice* -20 to +19, default 0 (lower = higher prio)
    - in Linux, a control over the *proportion of timeslice* (other Unixes: *absolute timeslice*)
    - `nice` syscall:
      ```
      #include <unistd.h>
      int nice(int inc);
      ```
  - *real-time priority* 0 to 99
- Timeslice / scheduler
  - Linux's CFS - Completely Fair Scheduler
    - does not directly assign timeslices to processes
    - assigns processes a proportion of the processor.
    - thus, amount of processor time a process receives is a function of the load of the system

[1] https://www.oreilly.com/library/view/linux-kernel-development/9780768696974/