* **Process**: A running instance of a program.  
   Example: Running firefox & starts a new process in the background.
* **Process States**:
  + **New** – The process is created but hasn’t started execution yet.
    - Example: When you type ./a.out in the terminal, the process is created
  + **Ready** – The process is waiting in the queue to be executed by the CPU.
    - Example: If programs are open, they all sit in the Ready queue, waiting for the CPU.
  + **Running** – The process is currently being executed by the CPU.
    - Example: A command like ls runs briefly in the CPU before finishing.
  + **Waiting (Blocked)** – The process is waiting for an event to occur (like input from a user).
    - Example: A process running read myfile.txt waits until the file is available.
  + **Terminated (Exit)** – The process has finished execution or was forcefully stopped.
* **Contents of a Process**: Code, Data, Stack, Heap, and Registers.
  + **Code Segment** – Contains the compiled instructions of the program (binary machine code).
    - Example: The compiled version of int main() { return 0; } is stored here.
  + **Data Segment** – Stores global and static variables.
    - Example: int global\_var = 10; // Stored in Data Segment
  + **Stack** – Stores function calls, local variables, and return addresses.
    - Example:

void myFunc() {

int x = 5; // Stored in the stack

}

Each function call creates a new stack frame.

* + **Heap** – Stores dynamically allocated memory (via malloc() or new).
    - Example:

int \*ptr = (int \*)malloc(sizeof(int)); // Allocated in Heap

This stays until freed with free().

* + **Registers** – Small, fast storage inside the CPU used to keep temporary values (like counters, instruction pointers).
    - Example: When adding two numbers, the values might be stored in registers before writing back to memory.
* **Types of CPU Scheduling**:
  + First-Come, First-Served (FCFS)
    - Processes are executed in the order they arrive.
    - Analogy**:** Like waiting in line at a coffee shop.
  + Shortest Job First (SJF)
    - The process with the smallest burst time executes first.
    - Analogy**:** The professor grading **shortest** assignments first.
  + Priority Scheduling
    - Each process has a priority, and the highest priority process runs first.
    - Analogy**:** Emergency patients in a hospital get treated first.
  + Round Robin (RR)
    - Each process gets a fixed time slice (quantum), then goes to the back of the queue.
    - Analogy**:** Sharing turns in a multiplayer game.
* **Multi-user Systems**: OS allowing multiple users to access simultaneously.
* **Multi-tasking Systems**: OS executing multiple processes concurrently.
* **Virtual Machine**: Software that emulates hardware to run OS instances.
* **Shell**: Command-line interface to interact with OS.
* **Commands**:
  + rm - Remove files (rm file.txt)
  + cd .. - Move to parent directory
  + cd ~ - Move to home directory
  + mkdir - Create a directory
  + rmdir - Remove an empty directory
  + ls - List directory contents
  + ls -l - List detailed directory contents
  + man - Show manual of a command
  + touch - Create an empty file or update timestamp
  + cat - View file contents
  + cp - Copy files
  + pwd - Print current directory
  + passwd - Change password
  + diff - Compare two files
  + wc - Count words, lines, characters in a file
  + chgrp - Change file group
  + kill - Terminate a process
  + date - Display system date and time
  + cal - Show calendar
  + df - Show disk usage
  + du - Show directory size
  + whatis - Get brief description of a command
  + which - Locate an executable
  + whereis - Find the location of binaries, source, and manual
  + find - Search for files
  + mv - Move or rename files
  + echo - Print text to terminal
  + read - Read user input
  + tail - Show last lines of a file
  + head - Show first lines of a file
  + uname - Show system information
  + uname -a - Show all system information
  + who - Show logged-in users
  + whoami - Show current user
  + random - Generate a random number
  + shuffle - Randomly reorder input
  + grep - Search text patterns
  + history - Show command history
  + fc - Edit and re-execute previous commands
  + aliases - Shortcuts for commands
  + cut - Extract specific text columns