Today Agenda :-

process & Threads

Concurrency vs Parallelism

Multicore system

Code

- .dmg . exe . apk . Rpm

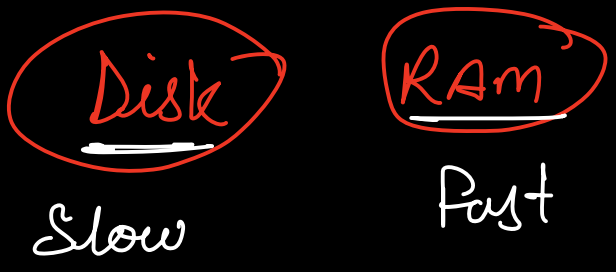
↳ executable files (Rom)

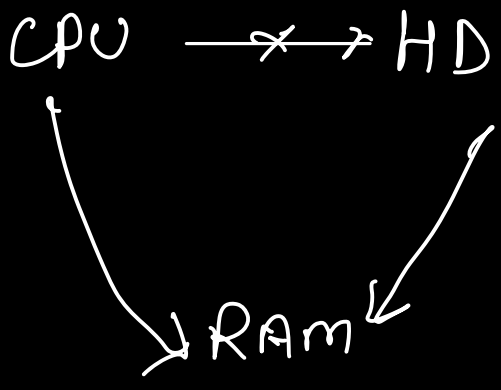
Code → Compile → executable file → download / install → (Rom) On disk

Program in execution ⇐ Process

SSD
HDD

Speed
Rom





1 TB → 1024 GB

↳ 4-5K → 8GB ≃ 10 GB user

Process :- Program in execution

Process Control Block (PCB) :-

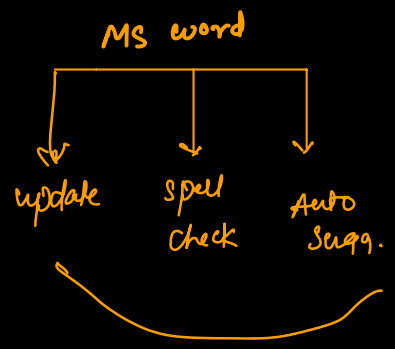
data structure that stores the info about a process

- pid
- list & variable
- Registers (typ. of memory)
- Priority
- Memory details
- Program Counter } → next line of code to be executed.
- Call stack

MS word :-

- Auto saving
- Spell checker
- Auto Correct / Grammar check
- Formatting
- UI updates
- Auto Suggestion

→ Diff Process



Process that run parallely

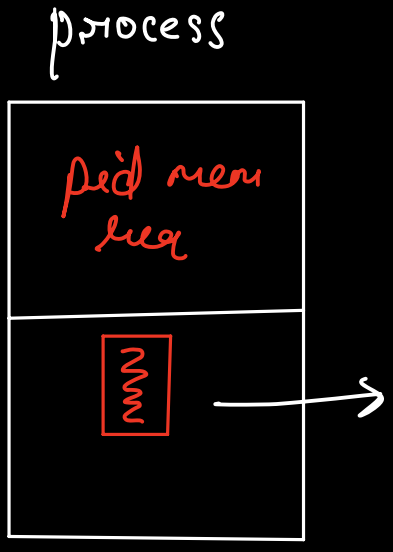
Thread :-

↓

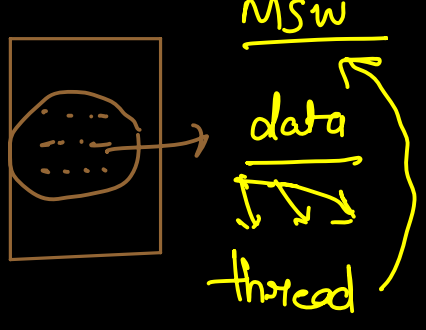
Unit of CPU execution

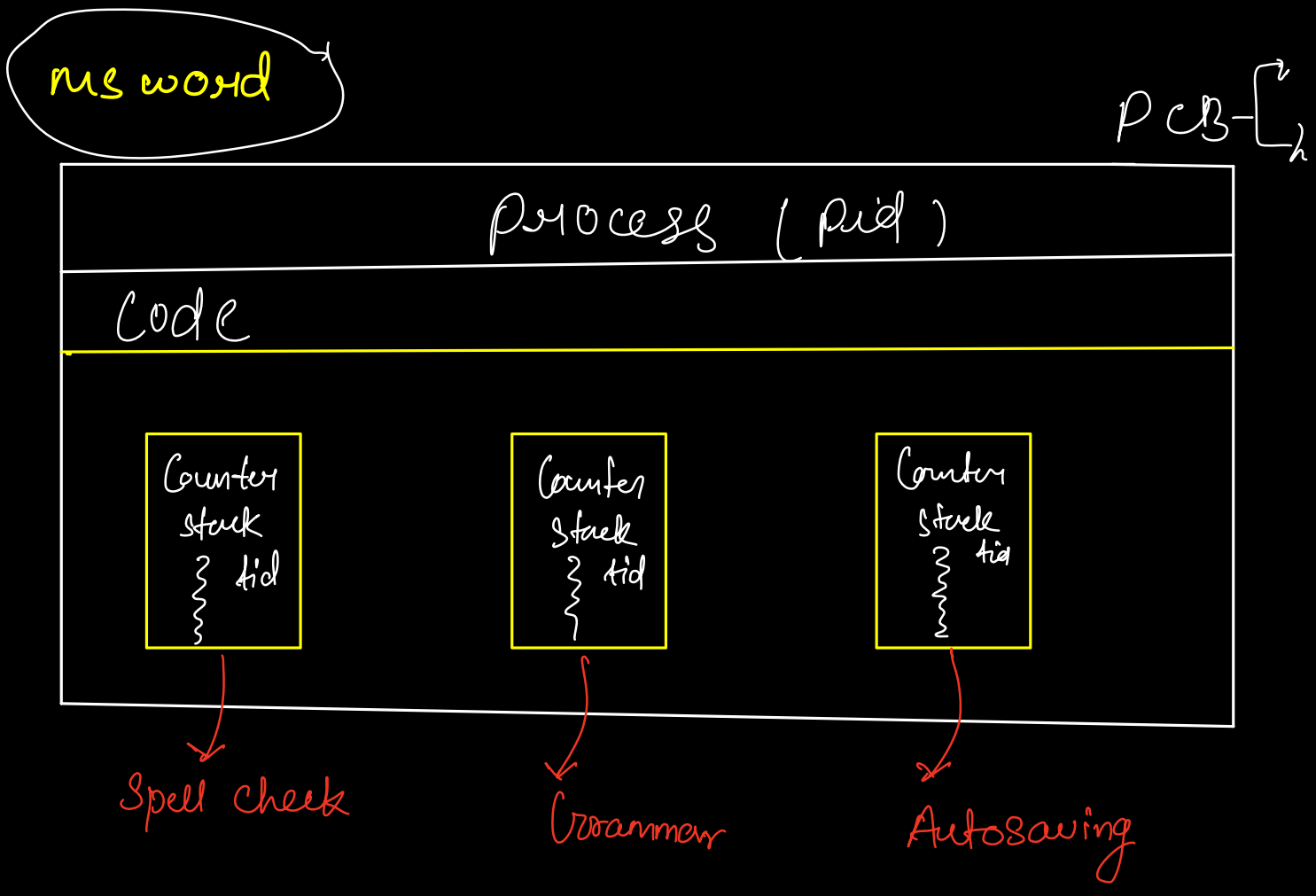
CPU executes thread

Whenever anything happens on your machine, there is a CPU running a thread which is running the code.



→ task is given to the CPU.





↳ All threads have access to the same data.

1) Memory :- Creation of thread has lower overhead. (easier)

2) Data Sharing :- All threads shares each other's data.

2000+ thread

OS → CPU Scheduler →

decides what threads to be executed

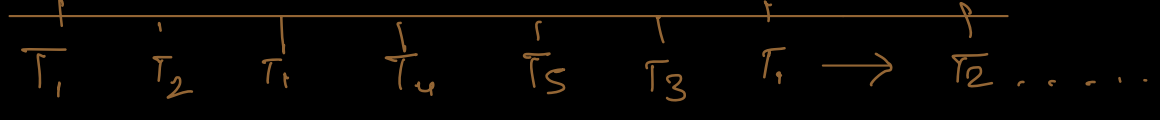
↳ priority

↳ Resources

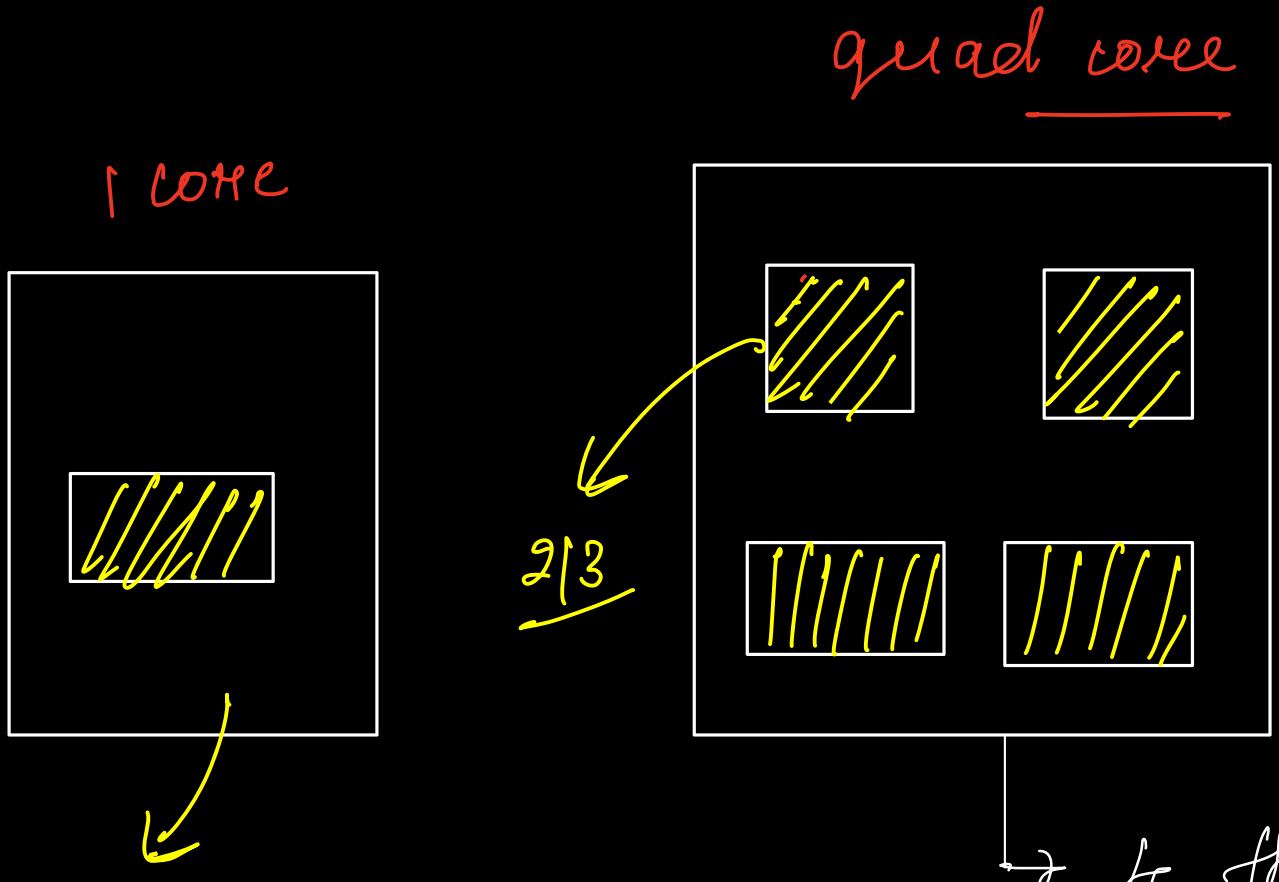
Algo

↳ time

OS Internals { SJF, SRTF, FCFS, Round Robin } CPU Scheduling Algo.



Single Core vs Multicore :-



chip inside CPU responsible for executing

→ 4 threads exactly at the same time

Each core can execute one thread at a time.

i3 → dual core

i5 → 4 core

i7 → quad core + Hyperthreading → 1 core might execute > 1 thread at a given point of time

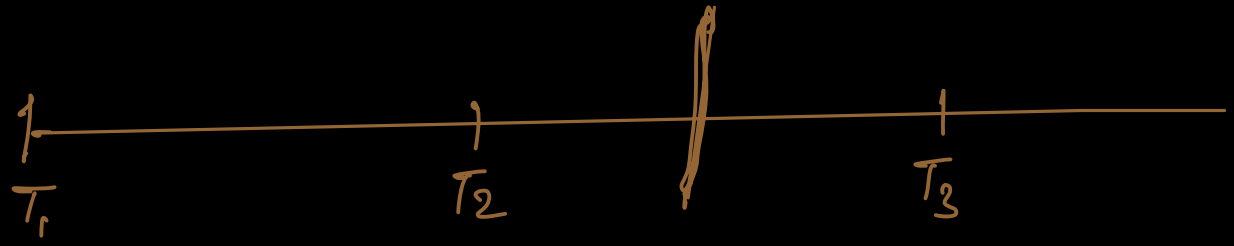
Concurrency vs Parallelism :-

Case 1

single core :-

CPU

Until one thread has completed its work, another thread not start



① How many threads will in practically completed ?

$\Rightarrow 1$

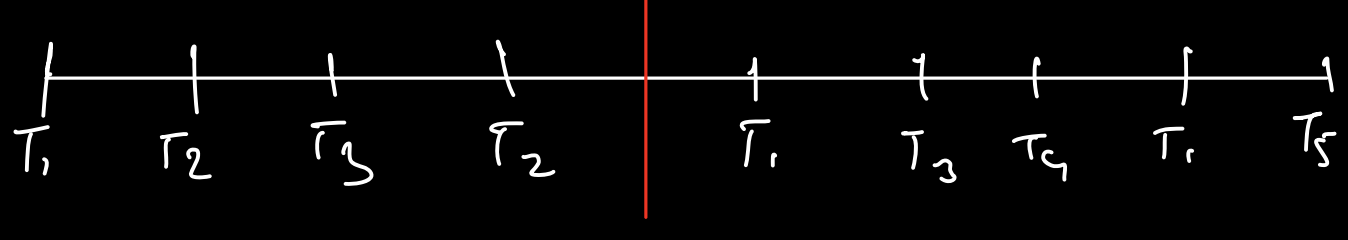
② How many threads are making progress ?

$\Rightarrow 1$

Case II Single Core CPU

Switch b/w thread is allowed

Context Switch



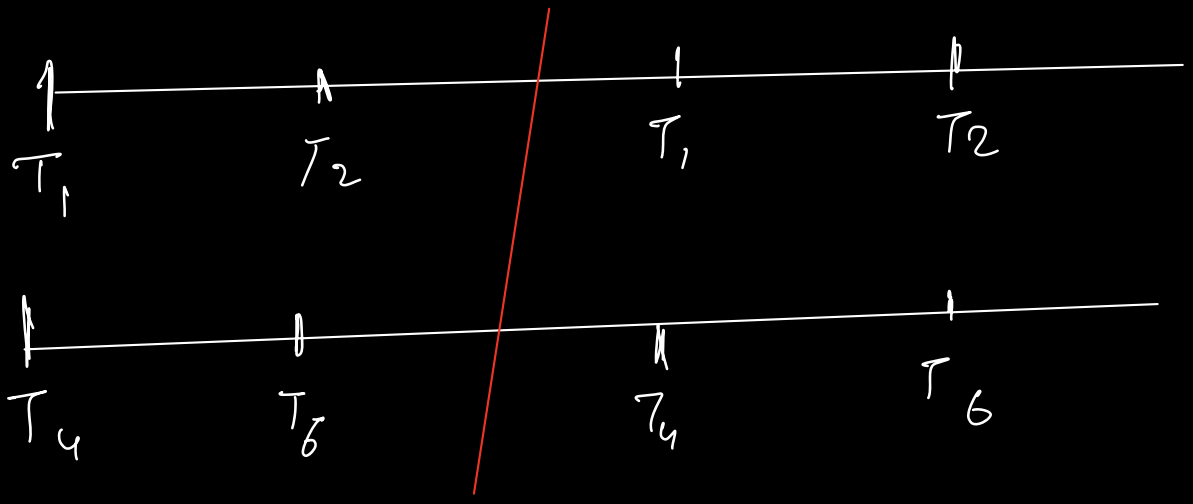
(1) How many threads will be in partially completed ?

>= 1

(2) How many threads are making progress ?

1

Case III Multicore (switch is also allowed)

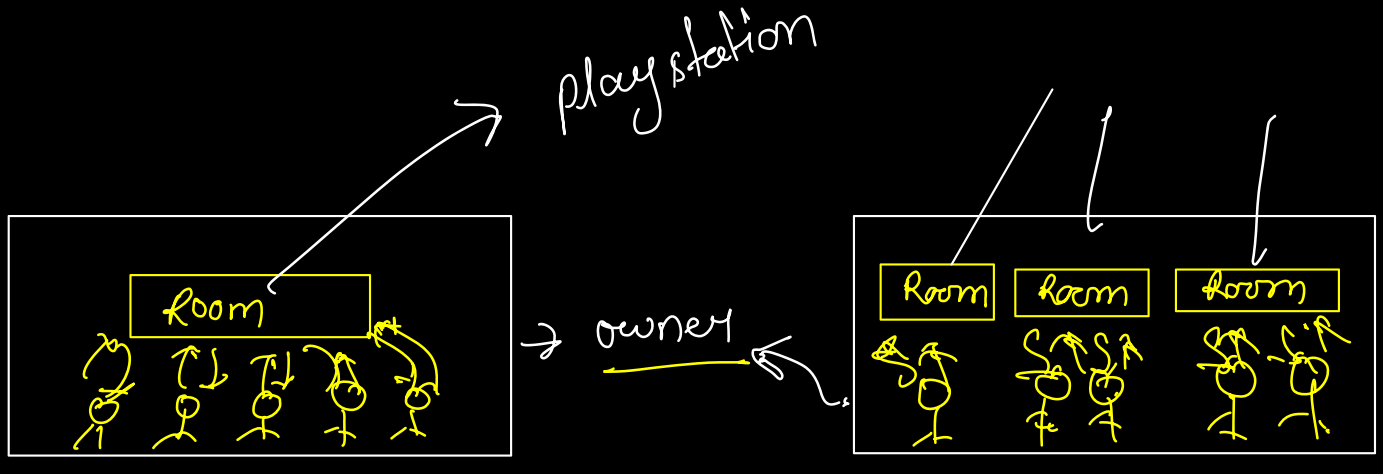


① How many threads will be in partially completed ?

>= 1 → Concurrency

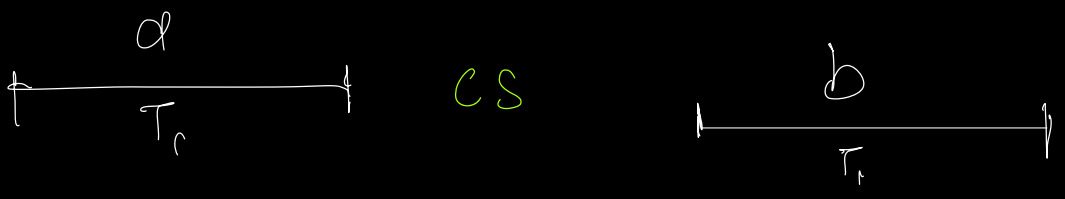
② How many threads are making progress ?

> 1 → parallelism

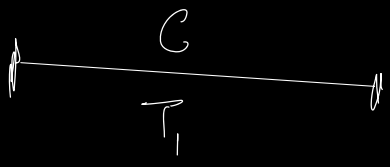


Concurrency VS parallelism

two session + 2.5 hr



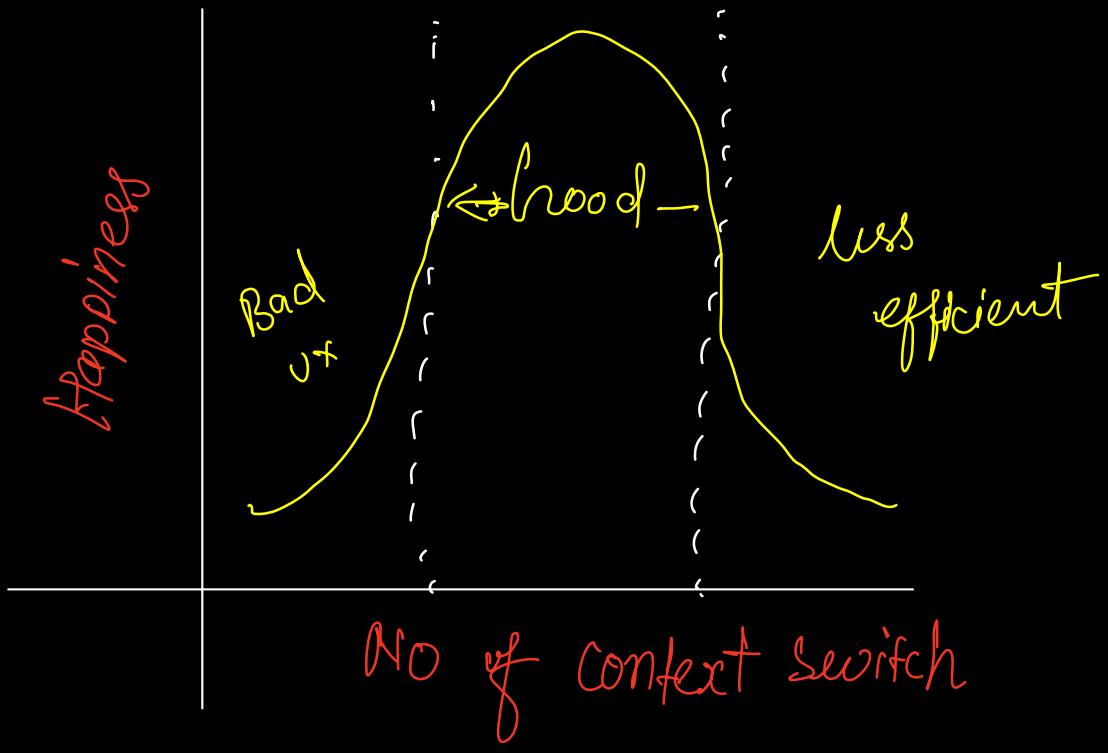
VS



1) $a+b > c$

2) $a+b = c$

3) $a+b < c$



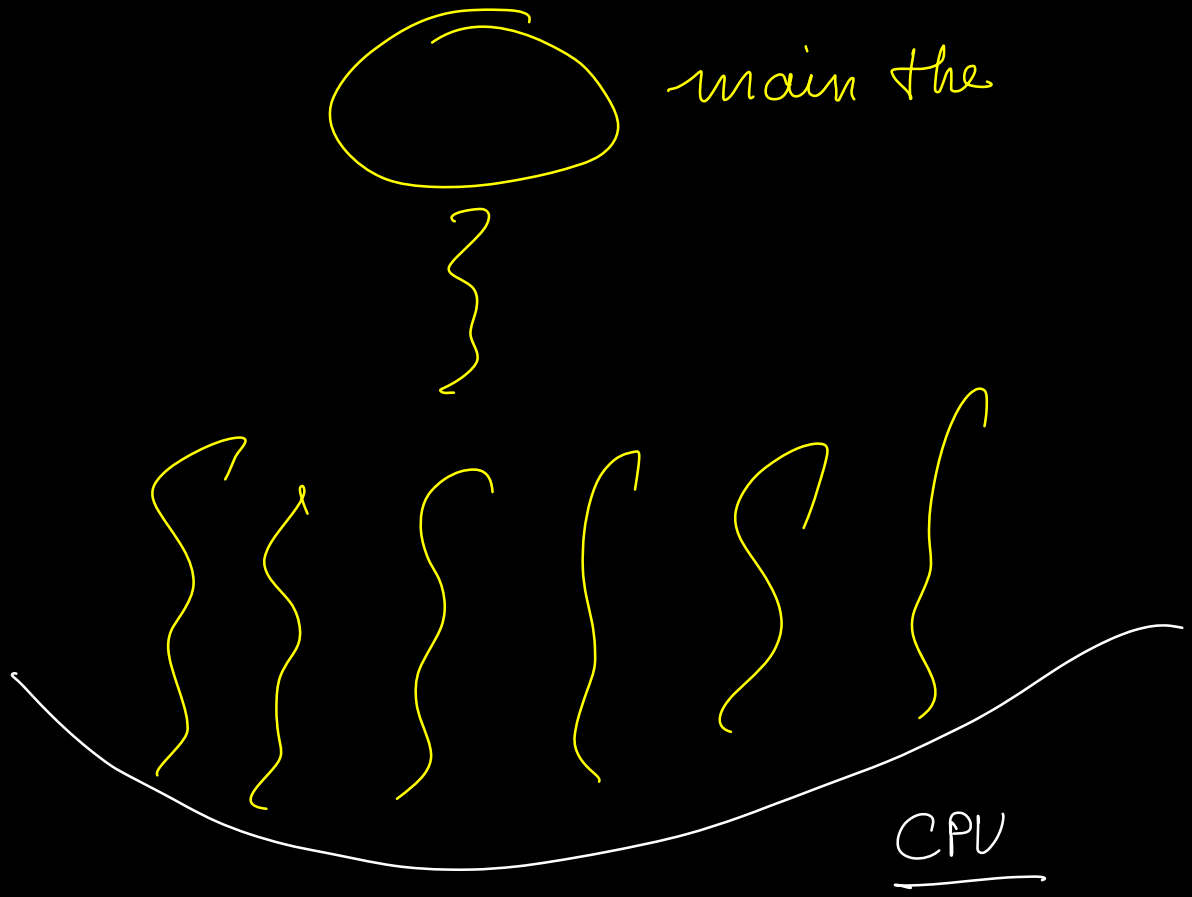
Execution of Threads :-

↳ Don't think of what thread to be created, always think about what task has to be done

1) Define the task
   ↳ Create a class for that task

2) your task / class should implements Runnable interface
   ↓
   class — implement Runnable {
   ↓ run() {
   ↓ }

3) In your task class, write the code that you want to execute in run() method.



Order is decided by the CPU not guaranteed

Undeterministic