multithreading and multiproccessing

Program

A **program** is a set of instructions written in a programming language that a computer can understand and execute to perform a specific task or solve a problem.

In simple terms:

A program tells the computer **what to do** and **how to do it**.

Programs can be:

- Small (like a calculator app)
- Large (like an operating system or a website)

Process

A **process** is an instance of a program that is currently being executed by a computer.

In simple terms:

- A **program** is just a file with code (instructions).
- A **process** is what happens when that program **runs**.

Key Points:

- Each process has its own memory, variables, and system resources.
- You can run the same program multiple times, and each run creates a separate **process**.

Example:

If you open **Notepad** three times on your computer:

- Notepad is the **program**
- Each open window is a separate **process**

Threads

A **thread** is the smallest unit of execution within a **process**.

In Simple Terms:

- A **program** becomes a **process** when it runs.
- A **process** can have **multiple threads** running inside it.

Each **thread** shares:

- The **same memory** and data of the process
- But runs **independently**, performing different tasks at the same time

Imagine a **restaurant** (process):

- **Chef**, **Waiter**, and **Cleaner** are all **threads**.
- They work in parallel, sharing the same kitchen and space (memory), but doing different jobs.

🧠 **Key Differences:**

Aspect	Process	Thread
Memory	Has its **own** memory	Shares memory with other threads
Overhead	Higher	Lower
Communication	Slower (inter-process)	Faster (shared memory)