1. what is the difference between a program and a process

A **program** and a **process** are related but represent different concepts in computing:

1. Program:

- **Definition**: A program is a **static set of instructions** written in a programming language. It is essentially a file or set of files that contain code and resources, stored on disk.
- **State**: A program is **passive**; it does not perform any actions until it is executed. It resides in storage (e.g., hard drive, SSD) and does not have an active state.
- **Example**: A text editor program or a browser application file on your computer.

2. Process:

- **Definition**: A process is an **instance of a program** that is currently being executed. It is an active entity, with its own execution state and allocated resources (e.g., memory, CPU time).
- **State**: A process is **dynamic** and has attributes such as process ID, current state (running, waiting, etc.), program counter, registers, and memory.
- **Lifecycle**: A process is created when a program is executed and is terminated when it finishes execution or is killed.
- **Example**: When you open your browser, it becomes a process, with memory allocated for it and running in your computer's RAM.

Key Differences:

- **Nature**: A program is static, while a process is dynamic.
- **State**: A program doesn't have an execution state, but a process has states like running, waiting, or terminated.
- **Existence**: A program can exist indefinitely on storage, while a process only exists as long as it is running.

In short, a program is the code; a process is what happens when that code is executed.

2. **Process state Example**

- You start your music player app (**New** state).
- The app is initialized and loaded into memory (**Ready** state).
- The scheduler gives the CPU to the music player, and it starts playing a song (**Running** state).
- The app needs to load the next song, so it issues an I/O request and moves to **Waiting**.
- The I/O operation completes, and the app goes back to the **Ready** state.
- The CPU assigns it time again, and it resumes playing the song (**Running**).
- When you close the app, it moves to the **Terminated** state.

3. PCB content example

- **Content**:

- **Process ID (PID)**: A unique identifier for the process (e.g., PID = 1001).
- **Process State**: The current state of the process (e.g., Running).
- **Program Counter (PC)**: The address of the next instruction to be executed (e.g., PC = 0x004000A4).
- **CPU Registers**: Values of registers used by the process (e.g., EAX = 5, EBX = 10).
- **Memory Management Information**: Information like base and limit registers, page tables, or segment tables.
- **I/O Status Information**: Details about files being used by the process, open file descriptors, etc.
- **Priority**: The process's priority level (e.g., Priority = 3).
- **Accounting Information**: CPU usage, time limits, etc.