WEEK-04

Contents:

Process Management

- Process, daemon
- Process states, PCB
- Process scheduling
- Queue Operations on Processes - Process creation, Process termination, Inter process communication.
- Scheduling - Long term, short term, and medium term;
- Context switch;
- Different types of CPU Schedulers (Basic concept),Process priority;
- Debugging (system hang)

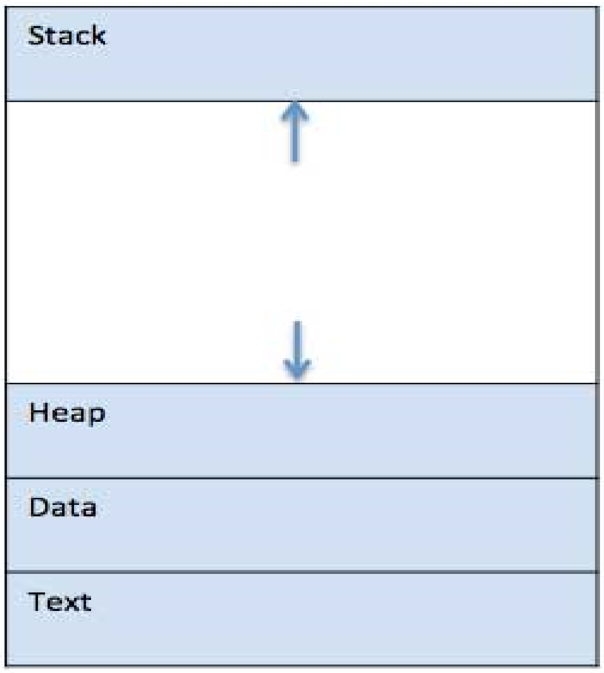
Introduction:

A process is a program in execution. The execution of a process must progress in a sequential fashion.

A process is defined as an entity, which represents the basic unit of work to be implemented in the system.

To put it in simple terms, we write our computer programs in a text file and when we execute this program, it becomes a process, which performs all the tasks mentioned in the program.

When a program is loaded into the memory and it becomes a process, it can be divided into four sections ─ stack, heap, text and data. The following image shows a simplified layout of a process inside main memory –



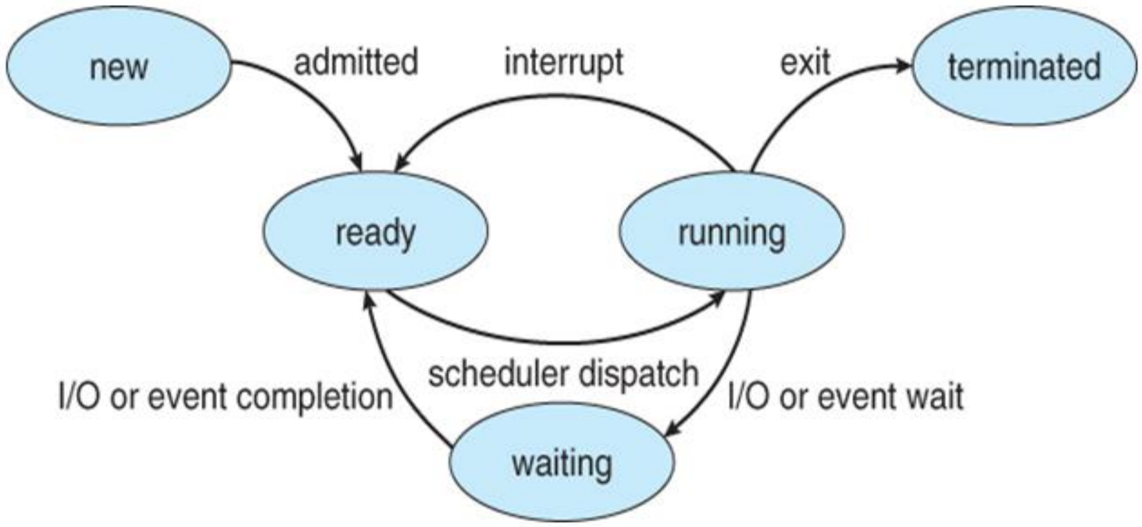
Daemons:

- ✓ **Daemons are processes that run unattended**.
- ✓ They are constantly run in the background and are available at all times.
- ✓ Daemons are usually started when the system starts, and they run until the system stops. A daemon process typically performs system services and is available at all times to more than one task or user

What is a daemon process Linux?

- ➢ A daemon is also called a background processes.
- ➢ It is a UNIX or Linux program that executes inside the background.
- ➢ Almost every daemon contains names that finish with "d" the letter.
- ➢ For example, sshd, this manages connections of SSH remote access, or the httpd daemon that manages the Apache server

Process States:



1. **New:**
   A program that is going to be picked up by the OS into the main memory is called a new process.

2. **Ready:**
   - Whenever a process is created, it directly enters to the ready state, in which, it waits for the CPU to be assigned.
   - The OS picks the new processes from the secondary memory and put all of them in the main memory.
   - The processes which are ready for the execution and reside in the main memory are called **ready state processes**.

3. **Running:**
   - One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm.
   - Hence, if we have only one CPU in our system, the number of running processes for a particular time will always be one.
   - If we have n processors in the system then we can have n processes running simultaneously.

4. **Block or wait:**
    - From the Running state, a process can make the transition to the block or wait state depending upon the scheduling algorithm or the intrinsic behavior of the process.
    - When a process waits for a certain resource to be assigned or for the input from the user then the OS move this process to the block or wait state and assigns the CPU to the other processes.

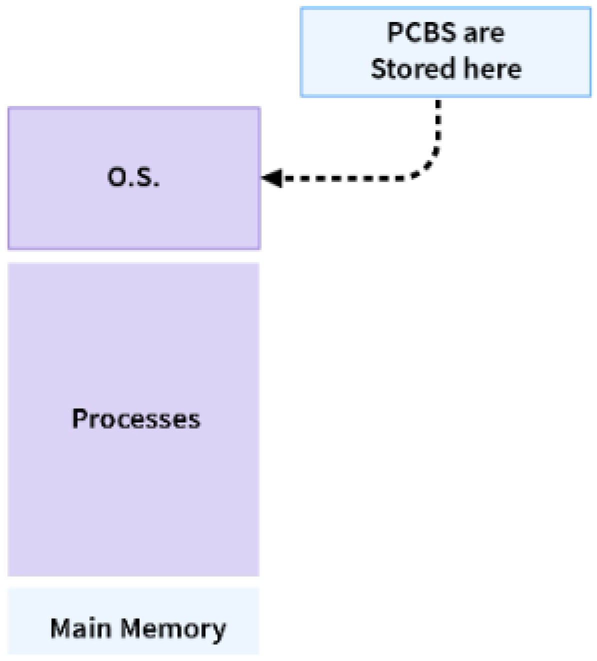
5. **Completion or termination:**
    - When a process finishes its execution, it comes in the termination state.
    - All the context of the process (Process Control Block) will also be deleted the process will be terminated by the Operating system.

6. **Suspend ready:**
    - A process in the ready state, which is moved to secondary memory from the main memory due to lack of the resources (mainly primary memory) is called in the suspend ready state.
    - If the main memory is full and a higher priority process comes for the execution then the OS have to make the room for the process in the main memory by throwing the lower priority process out into the secondary memory.
    - The suspend ready processes remain in the secondary memory until the main memory gets available.

Process Control Block

When the process is created by the operating system, it creates a data structure to store the information of that process. **This is known as Process Control Block (PCB).**

Process Control block (PCB) is a data structure that stores information of a process.



PCBs are stored in especially reserved memory for the operating system known as kernel space.

Role of Process Control Block

The role of the process control block arises as an identification card for each process. The Operating System does not know which process is which, until Operating System refers through the PCB of every process.

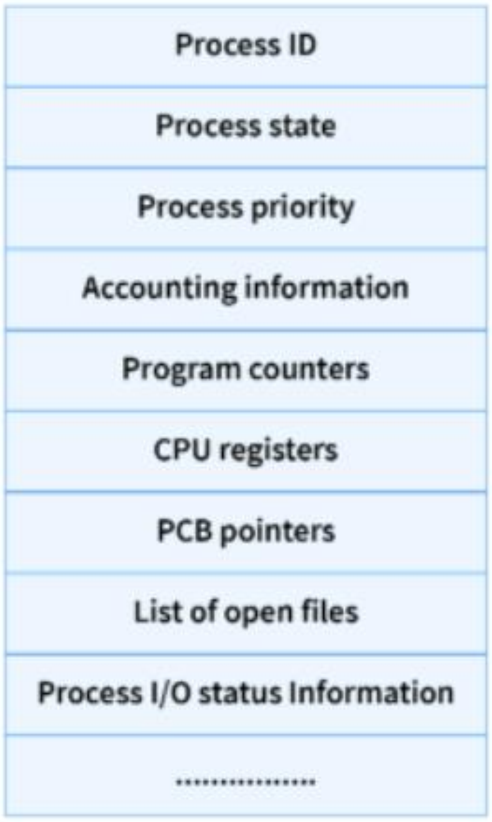
For Example: There are MS word processes, pdf processes, printing processes, and many background processes are running currently on the CPU. How will OS identify and manage each process without knowing the identity of each process?

Therefore, here PCB comes into play as a data structure to store information about each process.

Hence, whenever a user triggers a process (like print command), a process control block (PCB) is created for that process in the operating system which is used by the operating system to execute and manage the processes when the operating system is free.

Structure of Process Control Block

The process control block contains many attributes such as process ID, process state, process priority, accounting information, program counter, CPU registers`, etc for each process.



Process Scheduling

It is the activity of the OS, which removes the running process from the CPU and selects another process based on a particular strategy.

Process scheduling is an essential part of Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

Process Scheduling Queues

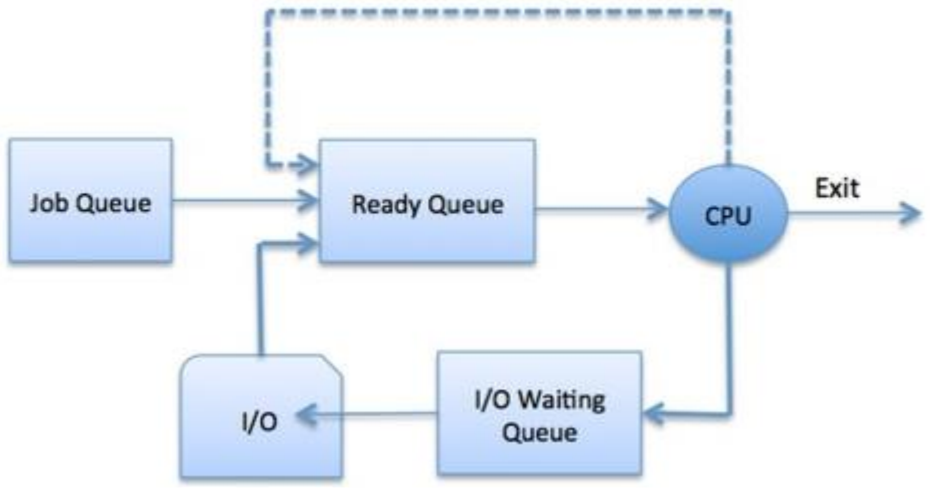
The OS maintains all PCBs in Process Scheduling Queues.

The OS maintains a separate queue for each of the process states and PCBs of all processes in the same execution state are placed in the same queue.

When the state of a process is changed, its PCB is unlinked from its current queue and moved to its new state queue.

The Operating System maintains the following important process scheduling queues −

- **Job queue** − this queue keeps all the processes in the system.
- **Ready queue** − this queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue.
- **Device queues** − the processes which are blocked due to unavailability of an I/O device constitute this queue.



The OS can use different policies to manage each queue (FIFO, Round Robin, Priority, etc.).
The OS scheduler determines how to move processes between the ready and run queues which can only have one entry per processor core on the system; in the above diagram, it has been merged with the CPU

Schedulers

Schedulers are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types −

1. Long-Term Scheduler
2. Short-Term Scheduler
3. Medium-Term Scheduler

1. Long Term Scheduler

- It is also called as **job scheduler**.
- A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.
- The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound.
- It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

2. Short Term Scheduler

- It is also called as CPU scheduler. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.
- Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

3. Medium Term Scheduler

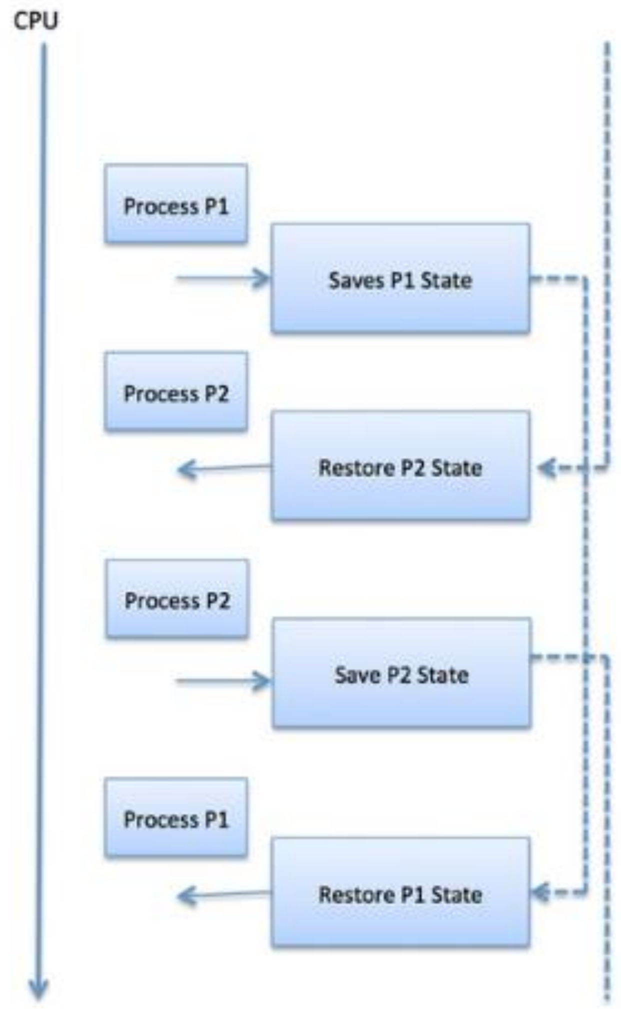
- Medium-term scheduling is a part of swapping.

- It removes the processes from the memory.
- It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.
- A running process may become suspended if it makes an I/O request.
- A suspended processes cannot make any progress towards completion.
- In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called swapping, and the process is said to be swapped out or rolled out. Swapping may be necessary to improve the process mix.

Context Switch

A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. Using this technique, a context switcher enables multiple processes to share a single CPU. Context switching is an essential part of a multitasking operating system features.

When the scheduler switches the CPU from executing one process to execute another, the state from the current running process is stored into the process control block. After this, the state for the process to run next is loaded from its own PCB and used to set the PC, registers, etc. At that point, the second process can start executing.



Context switches are computationally intensive since register and memory state must be saved and restored. To avoid the amount of context switching time, some hardware systems employ two or more sets of processor registers. When the process is switched, the following information is stored for later use.

- Program Counter
- Scheduling information
- Base and limit register value
- Currently used register
- Changed State
- I/O State information
- Accounting information

Operations on Process

The two main operations performed on Process are as follows −

Process Creation

There should be four principle events, which cause processes to be created.

1. **System initialization**

   Numerous processes are created when an operating system is booted. Some of them are −

- **Foreground processes** − Processes that interact with users and perform work for them.
- **Background processes** − It is also called as daemons and not associated with particular users, but instead has some specific function.

2. **Execution of a process-creation system call by a running process**

   The running process will issue system calls to create one or more new processes to help it do its job.

3. **A user request to create a new process**

✓ A new process is created with the help of an existing process executing a process creation system call.

✓ In UNIX, the system call which is used to create a new process is **fork()**

✓ In Windows, **CreateProcess()**, which has 10 parameters to handle both process creation and loading the correct program into the new process.

4. **Initiation of a batch job**

✓ Users are going to submit batch jobs to the system.

✓ When the operating system creates a new process and runs the next job from the input queue in it.

Process Termination

Process is terminated by a call to **kill** in UNIX or **Terminate Process** in windows.

Process is terminated due to following reason −

- **Normal exit** − most processes terminate when they have completed their work and execute a system call to exit.

- **Error exit** − the third type of error occurs due to program bugs like executing an illegal instruction, referencing, or dividing by zero.

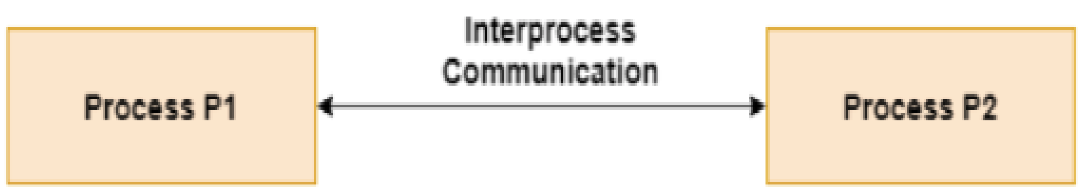
- **Fatal exit** − a termination of a process occurs when it discovers a fatal error.

- **Killed by another process** − a process executes a system call to kill some other process.

Inter-process Communication

Inter-process communication is the mechanism provided by the operating system that allows processes to communicate with each other.
This communication could involve a process letting another process know that some event has occurred or the transferring of data from one process to another.

A diagram that illustrates inter-process communication is as follows –



Process priority:

- ✓ Process with highest priority is to be executed first and so on.
- ✓ Processes with same priority are executed on first come first served basis.
- ✓ Priority can be decided based on memory requirements, time requirements or any other resource requirement.

- ✓ Process priority is simply the 'importance' of each process.
- ✓ Tasks that are essential for the smooth running of your computer (mostly system processes) are accorded a higher priority than an application running on top.

Debugging (system hang):

- ➢ Debugging is the process where the user needs to analyze the failure.
- ➢ And after seeing the failures they can check the performance and find out the error,
- ➢ After finding out the error and fixing all the error and this is called operating system debugging.