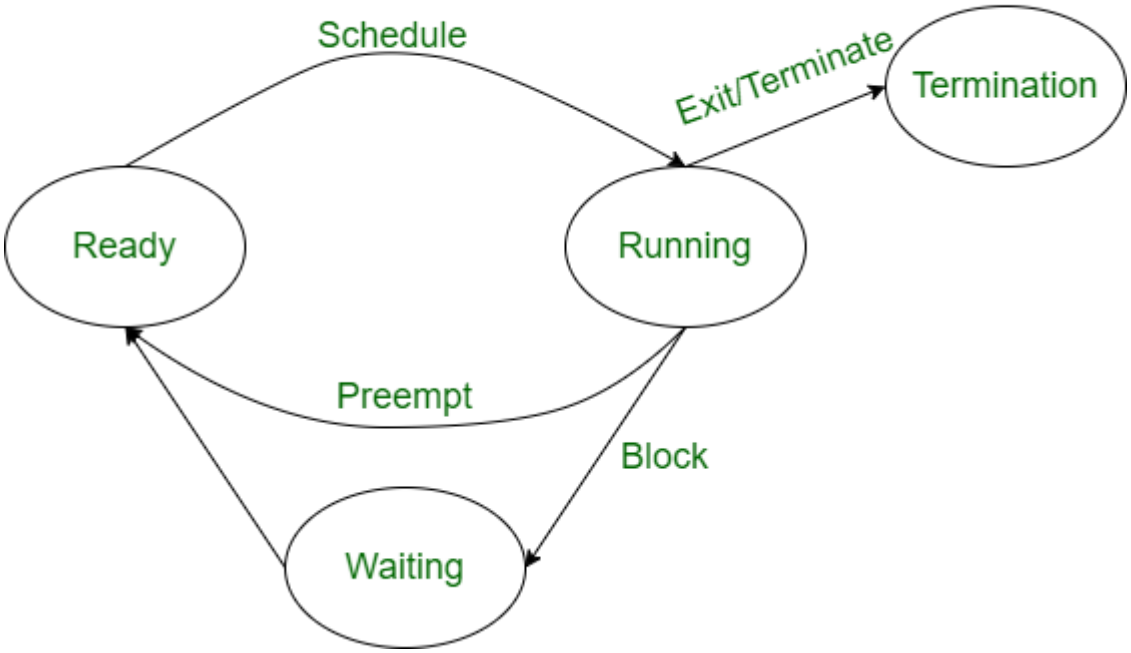
Operations on Processes

A process is an activity of executing a program. Basically, it is a program under execution. Every process needs certain resources to complete its task.

Operation on a Process

The execution of a process is a complex activity. It involves various operations. Following are the operations that are performed while execution of a process:



Creation

This is the initial step of the process execution activity. Process creation means the construction of a new process for execution. This might be performed by the system, the user, or the old process itself. There are several events that lead to the process creation. Some of the such events are the following:

1. When we start the computer, the system creates several background processes.
2. A user may request to create a new process.
3. A process can create a new process itself while executing.
4. The batch system takes initiation of a batch job.

Scheduling/Dispatching

The event or activity in which the state of the process is changed from ready to run. It means the operating system puts the process from the ready state into the running state. Dispatching is done by the operating system when the resources are free or the process has higher priority than the ongoing process. There are various other cases in which the process

in the running state is preempted and the process in the ready state is dispatched by the operating system.

Blocking

When a process invokes an input-output system call that blocks the process, and operating system is put in block mode. Block mode is basically a mode where the process waits for input-output. Hence on the demand of the process itself, the operating system blocks the process and dispatches another process to the processor. Hence, in process-blocking operations, the operating system puts the process in a 'waiting' state.

Preemption

When a timeout occurs that means the process hadn't been terminated in the allotted time interval and the next process is ready to execute, then the operating system preempts the process. This operation is only valid where CPU scheduling supports preemption. Basically, this happens in priority scheduling where on the incoming of high priority process the ongoing process is preempted. Hence, in process preemption operation, the operating system puts the process in a 'ready' state.

Process Termination

Process termination is the activity of ending the process. In other words, process termination is the relaxation of computer resources taken by the process for the execution. Like creation, in termination also there may be several events that may lead to the process of termination. Some of them are:

1. The process completes its execution fully and it indicates to the OS that it has finished.
2. The operating system itself terminates the process due to service errors.
3. There may be a problem in hardware that terminates the process.
4. One process can be terminated by another process.

Process Schedulers in Operating System

Process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

Process scheduling is an essential part of a Multiprogramming operating system. 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.

Categories in Scheduling

Scheduling falls into one of two categories:

- **Non-preemptive:** In this case, a process's resource cannot be taken before the process has finished running. When a running process finishes and transitions to a waiting state, resources are switched.

- **Preemptive:** In this case, the OS assigns resources to a process for a predetermined period of time. The process switches from running state to ready state or from waiting for state to ready state during resource allocation. This switching happens because the CPU may give other processes priority and substitute the currently active process for the higher priority process.

There are three types of process schedulers.

Long Term or Job Scheduler

It brings the new process to the 'Ready State'. It controls the ***Degree of Multi-programming***, i.e., the number of processes present in a ready state at any point in time. It is important that the long-term scheduler make a careful selection of both I/O and CPU-bound processes. I/O-bound tasks are which use much of their time in input and output operations while CPU-bound processes are which spend their time on the CPU. The job scheduler increases efficiency by maintaining a balance between the two. They operate at a high level and are typically used in batch-processing systems.

Short-Term or CPU Scheduler

It is responsible for selecting one process from the ready state for scheduling it on the running state. Note: Short-term scheduler only selects the process to schedule it doesn't load the process on running. Here is when all the scheduling algorithms are used. The CPU scheduler is responsible for ensuring no starvation due to high burst time processes. ***The dispatcher*** is responsible for loading the process selected by the Short-term scheduler on the CPU (Ready to Running State) Context switching is done by the dispatcher only. A dispatcher does the following:

1. Switching context.
2. Switching to user mode.
3. Jumping to the proper location in the newly loaded program.

Medium-Term Scheduler

It is responsible for suspending and resuming the process. It mainly does swapping (moving processes from main memory to disk and vice versa). Swapping may be necessary to improve the process mix or because a change in memory requirements has overcommitted available memory, requiring memory to be freed up. It is helpful in maintaining a perfect balance between the I/O bound and the CPU bound. It reduces the degree of multiprogramming.

Some Other Schedulers

- **I/O schedulers:** I/O schedulers are in charge of managing the execution of I/O operations such as reading and writing to discs or networks. They can use various algorithms to determine the order in which I/O operations are executed, such as FCFS (First-Come, First-Served) or RR (Round Robin).
- **Real-time schedulers:** In real-time systems, real-time schedulers ensure that critical tasks are completed within a specified time frame. They can prioritize and schedule

tasks using various algorithms such as EDF (Earliest Deadline First) or RM (Rate Monotonic).

Comparison among Scheduler

Long Term Scheduler	Short term schedular	Medium Term Scheduler
It is a job scheduler	It is a CPU scheduler	It is a process-swapping scheduler.
Generally, Speed is lesser than short term scheduler	Speed is the fastest among all of them.	Speed lies in between both short and long-term schedulers.
It controls the degree of multiprogramming	It gives less control over how much multiprogramming is done.	It reduces the degree of multiprogramming.
It is barely present or nonexistent in the time-sharing system.	It is a minimal time-sharing system.	It is a component of systems for time sharing.
It can re-enter the process into memory, allowing for the continuation of execution.	It selects those processes which are ready to execute	It can re-introduce the process into memory and execution can be continued.

Two-State Process Model Short-term

The terms “running” and “non-running” states are used to describe the two-state process model.

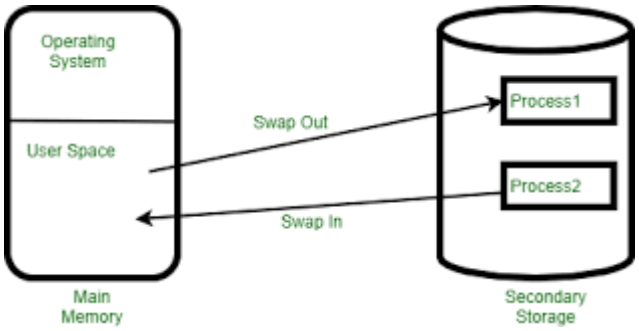
S.no	State Description
1.	**Running** A newly created process joins the system in a running state when it is created.

S.no	State Description
2.	**Not running** Processes that are not currently running are kept in a queue and await execution. A pointer to a specific process is contained in each entry in the queue. Linked lists are used to implement the queue system. This is how the dispatcher is used. When a process is stopped, it is moved to the back of the waiting queue. The process is discarded depending on whether it succeeded or failed. The dispatcher then chooses a process to run from the queue in either scenario.

Context Switching

In order for a process execution to be continued from the same point at a later time, context switching is a mechanism to store and restore the state or context of a CPU in the Process Control block. A context switcher makes it possible for multiple processes to share a single CPU using this method. A multitasking operating system must include context switching among its features.

The state of the currently running process is saved into the process control block when the scheduler switches the CPU from executing one process to another. The state used to set the PC, registers, etc. for the process that will run next is then loaded from its own PCB. After that, the second can start processing.



In order for a process execution to be continued from the same point at a later time, context switching is a mechanism to store and restore the state or context of a CPU in the PCB. A context switcher makes it possible for multiple processes to share a single CPU using this method. A multitasking operating system must include context switching among its features.

- Program Counter
- Scheduling information
- The base and limit register value
- Currently used register

- Changed State
- I/O State information
- Accounting information