Operating Systems

Process State :-

=> A program in execution is known as a process.

=> As a process executes, it changes state. The state of a process is defined in part by the current activity of that process.

=> Each process may be in one of the following states :-

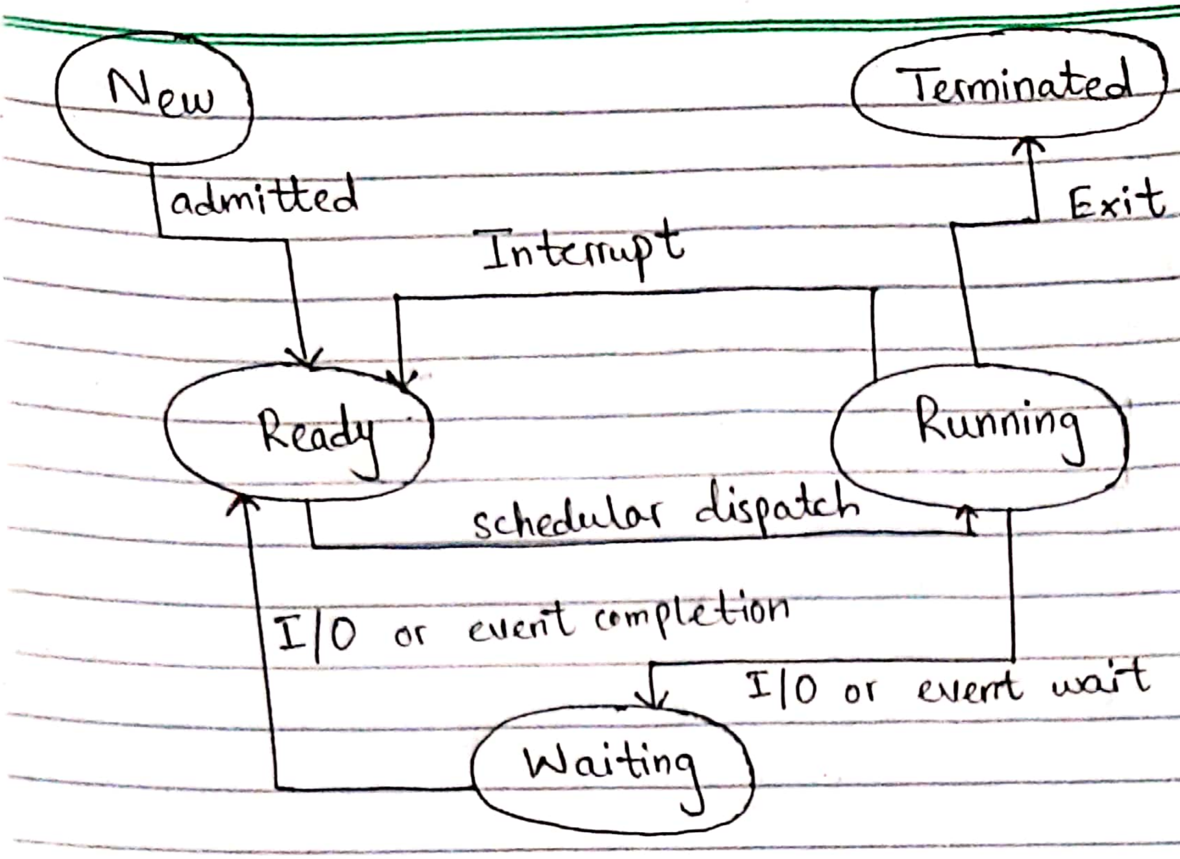
① New :- The process is being created.

② Running :- Instructions are being executed.

③ Waiting :- The process is waiting for some event to occur. (Such as an Input/Output completion or reception of a signal).

④ Ready :- The process is waiting to be assigned to a processor.

⑤ Terminated :- The process has finished execution



⇒ When a process is created in the new state, it stays there for some time and when the creation is fully completed, it is then moved to the Ready state.

⇒ In the ready state, the process is ready to begin execution and it is waiting to be assigned a processor so that it can begin its execution. The schedular dispatches it to the processor ~~in the running state~~ and when the process begins its execution then it is in the running state

⇒ In running state there are three ~~several~~ possibilities. One is that it was running so it finished its execution and was terminated successfully. Second is when the process was in ~~a~~ the state of execution, a new process came with high priority ~~on~~ so the former process was interrupted and goes back into the ready state. Once the interrupt has been handled the process is again dispatched to the running state. Third, while running the process needs an input/output operation, so it goes to the waiting state.

=) In waiting state, the process is waiting for input/output completion and remains there ~~to~~ till I/O is done. After the I/O operation is completed, the process moves back to the ~~rea~~ ready state where the schedular dispatches it to the running state again.

Process Control Block :-

=) Each process is represented in the operating system by a process control block.

=) An example diagram showing process control block:-

Process state
Process number
Program counter
Registers
Memory limits
List of open files
.....

⇒Following things are present in the process control block :-

① Process number :- shows the unique number or unique ID of a particular process. It helps in identifying a particular process.

② Process state :- tells us in which state a process exists at that particular moment.

③ Program counter :- indicates the address of the next instruction that has to be executed for that particular process.

④ CPU Registers :- tells us the registers that are being used by a particular process

5) CPU Scheduling information :- contains the priority of the processes, contains the pointer scheduling qeue (Scheduling determines which process has to be executed first).

) Memory management information :- represents the memory that is being used by a particular process.

7) Accounting information :- keeps an account of certain things like resources being used by a particular process (Resources = CPU, time, memory)

⑧ I/O status information :- represents the input/output devices that are being assigned to a particular process.

Process Scheduling :-

=) The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization.

=) The objective of time sharing is to switch the CPU among processes so frequently that users can interact with each program while its running.

=) To meet these objectives, the process scheduler selects an available process (possibly from a set of several available processes) for program execution on the CPU.

=) For a single-processor system, there will never be more than one running process.

=) If there are more processes, the rest will have to wait untill the CPU is free and can be rescheduled.

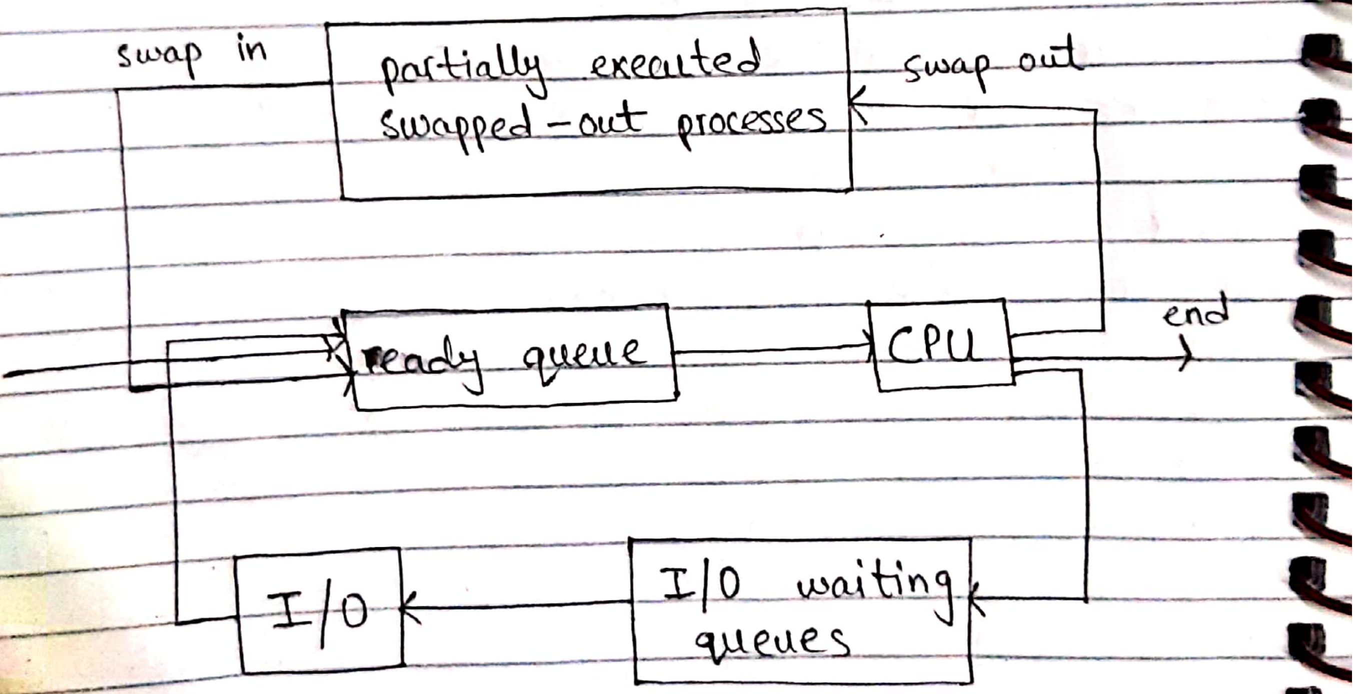
=) In order to help in process scheduling we have the scheduling queues.

⇒ Following are the types of scheduling queues :-

① Job queue :- As processes enter the system, they are put into a job queue, which consists of all processes in the system. It contains the list of all the processes that we have in the system.

② Ready queue :- The processes that are residing in main memory and are ready and waiting to execute are kept on a list called the ready queue.

⇒ The following diagram is of what happens when a process gets the CPU in the ready state;



=> From job queue, the processes come into the ready queue. When a process gets the CPU - there are several possibilities that exist.

=> One possibility is the process gets the CPU and completed its execution and ends.

=> Another possibility is when the process gets the CPU, another process comes with a higher priority, then the process with low priority get swapped out and is stored in the swapped-out list which then moves it into ready queue and then to the CPU for execution.

=> Another possibility is when the process gets the CPU, it requires an I/O operation. So, it is sent to the I/O queue. If the I/O devices are available then it will be sent to I/O device and after its operation is done then it will be sent to ready Queue and then to CPU. If I/O devices are not available then the process will be stored in I/O waiting queue.

Context Switch :-

=> Interrupts cause the operating system to change a CPU from its current task and to run a kernel routine.

=> Such operations happen frequently on general purpose systems.

Context = current state, current information

⇒ When an interrupt occurs, the system needs to save the current context of the process currently running on the CPU so that it can restore that context when its processing is done, essentially suspending the process and then resuming it.

⇒ Switching the CPU to another process requires performing a state save of the current process and a state restore of a different process. This task is known as context switch.

⇒ Context-switch time is pure overhead (because the system does no useful work while switching). When a CPU is switched from one process to another, at that moment only the switching part is working and no other ~~worl~~ useful work is done.

⇒ Its speed varies from machine to machine, depending on the memory speed, the no. of registers that must be copied, and the existence of special instructions.

⇒ Typical speeds are a few milliseconds.