**Q0**) What are the possible state transitions of a process?

Answer:

The process can be in any one of the following three possible states.  
1) Running (actually using the CPU at that time and running).  
2) Ready (runnable; temporarily stopped to allow another process run).  
3) Blocked (unable to run until some external event happens).

**Q1**) What are the differences between a thread and a process?

Answer:

| **Parameter** | **Process** | **Thread** |
| --- | --- | --- |
|  |  |  |
| Definition | Process means a program is in execution. | Thread means a segment of a process. |
| Lightweight | The process is not Lightweight. | Threads are Lightweight. |
| Termination time | The process takes more time to terminate. | The thread takes less time to terminate. |
| Creation time | It takes more time for creation. | It takes less time for creation. |
| Communication | Communication between processes needs more time compared to thread. | Communication between threads requires less time compared to processes. |
| Context switching time | It takes more time for context switching. | It takes less time for context switching. |
| Resource | Process consume more resources. | Thread consume fewer resources. |
| Treatment by OS | Different process are tread separately by OS. | All the level peer threads are treated as a single task by OS. |
| Memory | The process is mostly isolated. | Threads share memory. |
| Sharing | It does not share data | Threads share data with each other. |

**Q2**) What is a race condition?

Answer:

A race condition is an undesirable situation that occurs when a device or system attempts to perform two or more operations at the same time, but because of the nature of the device or system, the operations must be done in the proper sequence to be done correctly.

Race conditions are most commonly associated with computer science and programming. They occur when two computer program processes, or threads, attempt to access the same resource at the same time and cause problems in the system.

**Q3**) Five jobs are waiting to be run. Their expected run times are 9, 6, 3, 5, and *X*. In what order should they be run to minimize average response time? Given X = 10 and X = 1

• 0 < X ≤ 3: X, 3, 5, 6, 9

• 3 < X ≤ 5: 3, X, 5, 6, 9.

• 5 < X ≤ 6: 3, 5, X, 6, 9.

• 6 < X ≤ 9: 3, 5, 6, X, 9.

• X > 9: 3, 5, 6, 9, X.

**Q4**) Five batch jobs *A* through *E*, arrive at a computer center at almost the same time. They have estimated running times of 10, 6, 2, 4, and 8 minutes. Their (externally determined) priorities are 3, 5, 2, 1, and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the mean process turnaround time.

(a) Round robin (RR=4).

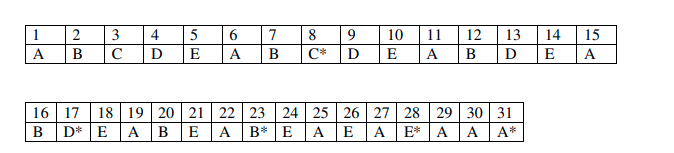
(b) Priority scheduling.

(c) First-come, first-served (run in order 10, 6, 2, 4, 8).

(d) Shortest job first.

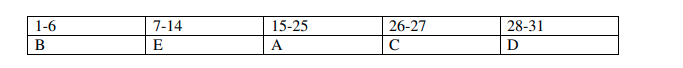
For (a), assume that the system is multiprogrammed, and that each job gets its fair share of the CPU. For (b) through (d) assume that only one job at a time runs, until it finishes. All jobs are completely CPU bound.

1. Round robin (RR=4).



Average turnaround = (8 + 17 + 23 + 28 + 31)/5 = 107/5 = 21.4 minutes

1. Priority scheduling.



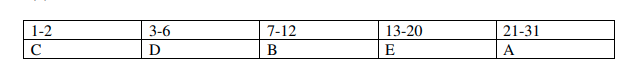
Avg. turnaround = (6 + 14 + 25 + 27 + 31)/5 = 103/5 = 20.6 minutes

1. First-come, first-served (run in order 10, 6, 2, 4, 8).



Avg. turnaround = (11 + 17 + 19 + 23 + 31)/5 = 101/5 = 20.2 minutes

1. Shortest job first.



Avg. turnaround = (2 + 6 + 12 + 20 + 31)/5 = 71/5 = 14.2 minutes

**Q5)** What is the difference between preemption and non-preemption in the context of process scheduling.

Answer:

Differences Between Preemptive and Non-Preemptive Scheduling:

In preemptive scheduling, the CPU is allocated to the processes for a limited time whereas, in Non-preemptive scheduling, the CPU is allocated to the process till it terminates or switches to the waiting state.

The executing process in preemptive scheduling is interrupted in the middle of execution when higher priority one comes whereas, the executing process in non-preemptive scheduling is not interrupted in the middle of execution and waits till its execution.

In Preemptive Scheduling, there is the overhead of switching the process from the ready state to running state, vise-verse and maintaining the ready queue. Whereas in the case of non-preemptive scheduling has no overhead of switching the process from running state to ready state.