Atishay Jain 2110385 21103285

Date.....

# Operating Systems Tutorial-3

Q1. There are 2 types of scheduling in context of OS→

1) Preemptive Scheduling → In this, the operating system can interrupt a running process & switch to another process based on priority. It leads to better responsiveness but potentially causing more frequent context switches.

2) Non-Preemptive Scheduling → It is also known as cooperative scheduling, allowing a running process to voluntarily release the CPU when it's done. It leads to longer response time but is simpler to implement.

Q2. CPU Utilization → The percentage of time the CPU is actively working on tasks. High utilization can indicate efficiency but excessive levels may lead to performance issues.

Throughput → The no. of tasks completed in a given time, reflecting system efficiency.

Turnaround Time → The total time a task takes to complete, considering queue time, exec. time & I/O wait time.

Waiting Time → The time a task spends in the ready queue before getting CPU time. Lower waiting time is better.

Load Average → A measure of system workload over different time intervals. High load average can suggest potential overloading.

Response Time → The time from task initiation to its start of execution. Low response time is crucial for system responsiveness.

Spiral

Date.....

Q3. Round Robin (RR) scheduling is well-suited for time shared operating systems due to its cyclic allocation of fixed time slices to processes. This ensures fairness, responsiveness, and efficient CPU sharing among multiple processes. This pre-emptive algorithm prevents any process from monopolizing the CPU and is effective for ~~effective of~~ interactive tasks in multi-user environments.

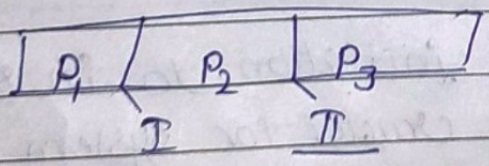
Q4 Shortest job first scheduling would result in maximum throughput as →

In this, the process having the shortest burst time will be executed first. It can be both pre-emptive & non preemptive, It reduces the waiting time for other process, which also results in max. throughput.

Q5. Using shortest remaining time first →

Process Table →

| Pid | AT | BT | CT |
|---|---|---|---|
| 1 | 0 | 10 | 10 |
| 2 | 2 | 20 | 30 |
| 3 | 6 | 30 | 60 |

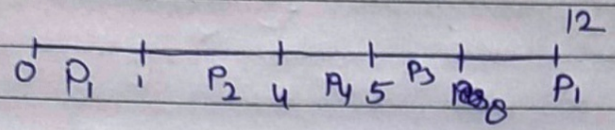



No. of context switches = 2

Date.....

Q6

| Process | AT | BT |
|---|---|---|
| $P_1$ | 0 | 5 |
| $P_2$ | 1 | 3 |
| $P_3$ | 2 | 3 |
| $P_4$ | 4 | 1 |

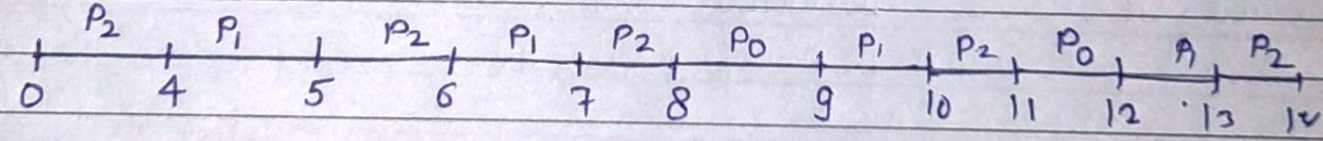



| Process | Turn around Time |
|---|---|
| $P_1$ | 12 |
| $P_2$ | 3 |
| $P_3$ | 6 |
| $P_4$ | 1 |

Average Turnaround Time = $\frac{12+3+6+1}{4} = 5.5$

Q7.

| P Id | AT | BT |
|---|---|---|
| 0 | 0 | 2 |
| 1 | 0 | 4 |
| 2 | 0 | 8 |




| PId | Turnaround Time |
|---|---|
| 0 | 12 |
| 1 | 13 |
| 2 | 14 |

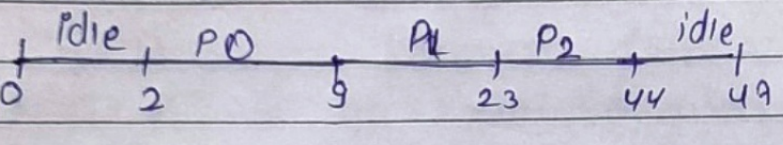
Avg Turnaround Time = $\frac{12+13+14}{3} = 13$

Date.....

Q8.

| PID | AT | IO | BT | IO |
|---|---|---|---|---|
| P0 | 0 | 2 | 7 | 1 |
| P1 | 0 | 4 | 14 | 2 |
| P2 | 0 | 6 | 21 | 3 |

P0 will spend 2 units in IO, then 7 in BT, then P1 will spend 14 units in BT (as 4 units of IO were spend when P0 was running), then P2 will spend 21 units in BT (as its 6 units of IO spend when prev. processes were running) & at last 3 units in I/o




Total time = 47
Total idle time = 5
Percent. of idle time = $\frac{5}{47} \times 100 = 10.6\%$.

Q9 P reads 40 sec.
Multilevel Feedback Queue:
Q1 = 2
Q2 = 7 (2+5)
Q3 = 12 (7+5)
Q4 = 17 (12+5)
Q5 = 2 (left time)

Process is interrupted 4 times & on 5th queue, the process completed its execution.

Spiral

Date.....

Q10. Dispatcher → Component that switches CPU bw processes by performing context switches, managing efficient resource allocation.

Short Term Scheduler → Selects processes from the ready queue for immediate execution, optimising CPU utilization and responsiveness.

Long Term Scheduler → Controls which new processes are admitted into memory, balancing system performance by managing the degree of multiprogramming.

Ready Queue → Holds processes ready for execution, allowing short-term scheduler to quickly choose tasks to run on the CPU.

Context Switch → Process of saving and restoring the state of a process during task switching, enabling multitasking and maintaining process execution continuity.