SJF algorithm can be pre-emptive or non-preemptive. The next CPU burst of newly arrived process may be shorter than what is left of the currently executing process. Pre-emptive SJF will preempt the currently running process.

Example:

Process	Arrival Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

Gantt Chart

P_1	P_2	P_4	P_1	P_3
0 – 1	1 – 5	5 – 10	10 – 17	17 – 26

The average waiting time is:

$$= [(10-1) + (1-1) + (17-2) + (5-3)]/4$$

$$= 26/4$$

$$= 6.5 ms.$$

Non-premptive would have resulted in 7.75ms.

Priority Scheduling

- SJF is a special case of priority scheduling.
- A priority is associated with each process & CPU is allocated to the process with highest priority
- Starvation low priority process may never execute and to solve that, as time progresses, we can increase the priority of the process. This process is called aging.

Example:

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

Gantt chart

P_2	P_5	P_1	P_3	P_4

0 1 6 16 18 19

Average waiting time $= 0+1+6+16+18 / 5$

$= 8.2$ ms.

prakash begade

- Priority scheduling can be preemptive or non preemptive
- Priorities can be defined internally or externally.

  Internal - time limits, memory requirements, number of open files, ratio of average I/O burst to average CPU burst etc.

  External - Importance of process, funds being paid for computer use, political etc.

Round-Robin Scheduling

- Designed for time sharing systems.
- Similar to FCFS but preemption is added
- We define time quantum or time slice & is generally from 10 to 100 ms in length.
- The ready queue is treated as Circular queue
- The CPU scheduler will ~~then~~ select the next process in ready queue after the time quantum.

Example:

Process	Burst Time
P_1	24
P_2	3
P_3	3

Time quantum = 4ms.

Gantt Chart

P1	P2	P3	P1	P1	P1	P1	P1

0 4 7 10 14 18 22 26 30

Avarage waiting time is

$P_1 = 0 + (10 - 4) = 6\,ms$

$P_2 = 4\,ms$

$P_3 = 7\,ms$

$= \frac{6+4+7}{3} = 5.66\,ms$

Example:

Process	Burst Time
P1	21
P2	3
P3	6
P4	2

Time Quantum = 5ms

Gantt Chart

P1	P2	P3	P4	P1	P3	P1	P1	P1

0 5 8 13 15 20 21 26 31 32

Waiting Time for,

$P_1 = 0 + (15-5) + (21-20) = 0 + 10 + 1$
$= 11ms$

$P_2 = 5ms$

$P_3 = 8 + (20-13) = 8 + 7 = 15ms$

$P_4 = 13ms$

Average Waiting Time =

$$= \frac{11+5+15+13}{4}$$

$$= \frac{44}{4} = 11ms.$$

- If there are n processes in the ready queue and the time quantum is q, then each process gets 1/n of the CPU time in chunks of at most q time units.

- Each process must wait no longer than $(n-1) * q$ time units ~~until~~ until its next time quantum.

- The performance of the algorithm depends on the size of the time quantum. If time quantum is large, it tends to become FCFS. If small there will be too many context switching.

- Example: for a process of time 10,

if Quantum $Q = 12$ → 0 Context Switches
$Q = 6$ 1 Context Switch
$Q = 1$ 9 Context Switches.

Multilevel Queue Scheduling

- Partitions the ready queue into several ready queues
- Processes are assigned to the queue, based on the properties
- Each queue can have its own algorithm among foreground and background processes.