**NAME – Abhay Singh Kushwaha SECTION – K21SP**

**REGISTRATION NUMBER – 12103254 ROLL NO- 40**

**COURSE – Operating System COURSE CODE – CSE316**

**GitHub link ---** [AbhaySinghkushwaha/OS\_PROJECT (github.com)](https://github.com/AbhaySinghkushwaha/OS_PROJECT/tree/main)

**DESCRIPTION OF PROBLEM:-**

This is a scheduling program to implement a Queue with two levels:

Level 1 : Fixed priority preemptive Scheduling

Level 2 : Round Robin Scheduling

For a Fixed priority pre-emptive scheduling if one process P1 is scheduled and running and another process P2 with higher priority comes. The New process with high priority process P2 preempts currently running process P1 and process P1 will go to second level queue. Time for which process will strictly execute must be considered in the multiples of 2.

All the processes in second level queue will complete their execution according to round robin scheduling.

In this program Queue 2 will be processed after Queue 1 becomes empty and Priority of Queue 2 has lower priority than in Queue 1.

**Fixed**-**priority preemptive scheduling**

Fixed-priority preemptive scheduling is a scheduling system commonly used in real-time systems. With fixed priority preemptive scheduling, the scheduler ensures that at any given time, the processor executes the highest priority task of all those tasks that are currently ready to execute.

**A lgorithm:**

In this program algorithm for round robin scheduling and multilevel queue scheduling is used.

**------ Algorithm For Multilevel Queue:**

When a process starts executing then it first enters queue 1.

In queue 1 process executes for 4 unit and if it completes in this 4 unit or it gives CPU for I/O operation in this 4 unit than the priority of this process does not change and if it again comes in the ready queue than it again starts its execution in Queue 1.

1. If a process in queue 1 does not complete in 4 unit then its priority gets reduced and it shifted to queue 2
2. Above points 2 and 3 are also true for queue 2 processes but the time quantum is 8 unit.In a general case if a process does not complete in a time quantum than it is shifted to the lower priority queue.
3. In the last queue, processes are scheduled in FCFS manner.
4. A process in lower priority queue can only execute only when higher priority queues are empty.
5. A process running in the lower priority queue is interrupted by a process arriving in the higher priority queue.

**Round Robin Scheduling:**

Round Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way.

* It is simple, easy to implement, and starvation-free as all processes get fair share of CPU.
* One of the most commonly used technique in CPU scheduling as a core.
* It is preemptive as processes are assigned CPU only for a fixed slice of time at most.
* The disadvantage of it is more overhead of context switching.

**------Algorithm for round robin scheduling:**

**How to compute below times in Round Robin using a program?**

1. Completion Time: Time at which process completes its execution.
2. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
3. Waiting Time(W.T): Time Difference between turn around time and burst time.  
   Waiting Time = Turn Around Time – Burst Time

**Steps to find waiting times of all processes:**

1. Create an array **rem\_bt[]** to keep track of remaining burst time of processes. This array is initially a

copy of bt[] (burst times array)

1. Create another array **wt[]** to store waiting times of processes. Initialize this array as 0.
2. Initialize time : t = 0
3. Keep traversing the all processes while if it is all processes are not done. Do following for i'th process

not done yet.

* 1. If rem\_bt[i] > quantum
     1. t = t + quantum
     2. bt\_rem[i] -= quantum;

1. Else // Last cycle for this process
   1. t = t + bt\_rem[i];
   2. wt[i] = t - bt[i]
2. bt\_rem[i] = 0; // This process is over

# Boundary Conditions:

|  |
| --- |
| * Level 1 : Fixed priority preemptive Scheduling |
| * Level 2 : Round Robin Scheduling |
| * Consider: 1. Queue 2 will be processed after Queue 1 becomes empty. |
| * Consider 2. Priority of Queue 2 has lower priority than in Queue 1. |

**Test Cases:**

* + Time Quantum for Fixed priority queue- 2
  + Time Quantum for Round Robin queue- 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst Time | Priority | Turnaround Time | Waiting Time |
| 1 | 0 | 4 | 1 | 10 | 6 |
| 2 | 0 | 3 | 2 | 13 | 10 |
| 3 | 0 | 8 | 1 | 8 | 3 |
| 4 | 10 | 5 | 1 | 20 | 12 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst Time | Priority | Turnaround Time | Waiting Time |
| 1 | 0 | 4 | 1 | 9 | 5 |
| 2 | 1 | 3 | 2 | 15 | 9 |
| 3 | 2 | 6 | 1 | 17 | 14 |
| 4 | 4 | 6 | 1 | 15 | 9 |