TSN2101 – Operating Systems

Assignment

Instructions:

Form a group with a maximum of **4 members** (you are allowed to form a group with less, minimum is 1 – you'll be doing the assignment alone).

You are allowed to use any programming language to simulate these concepts, not limited to C++, Java, Python, etc. Submit your program with documentation (zipped) to MMLS by **27 Sept 2019, 12pm**. You are to **present** your work during your tutorials in **Week 14**.

Any form of plagiarism will not be tolerated. No marks will be given for this assignment if plagiarism is detected.

Simulation of CPU scheduling algorithms

Perform implementation of ***all*** of the following **CPU scheduling algorithms** with a collection of processes, and also perform comparison of the given algorithms using different scheduling criteria.

- **FCFS (First Come First Served)-based pre-emptive Priority**:

Scheduling is based on priority. When two or more processes have the same priority, the process arrived first is selected.

If a new process arrives with a higher priority, it will pre-empt the running process. Priority ranges from 1 to 6 with smaller number indicates higher priority.

- **Round Robin Scheduling with Priority:**

Round Robin (RR) scheduling is the preemptive version of FCFS algorithm that selects the process that has been in the ready queue for the longest period of time. (i.e) the process is selected in round- robin manner. FCFS is used to select the process, after the expiry of specified time quantum. However, if a higher priority process comes into the FCFS queue, it goes to the head of the queue.

- **Three-level Queue Scheduling:**

A three level scheduling algorithm partitions the ready queue into two separate queues. Assume that the processes are permanently assigned to the queues based on priority. Assume that processes with priority number from 1 to 2 are assigned to queue 1, priority number 3 to 4 are assigned to queue 2 and the processes with priority number more than 4 are assigned to queue 3. Priority ranges from 1 to 6.

Each queue has its own scheduling algorithm. Processes within Queue 1 is scheduled by Round Robin (RR) algorithm while processes within Queues 2 and 3 are scheduled by First-Come-First-Serve (FCFS) algorithm.

Scheduling between queue 1, queue 2 and 3 is implemented as fixed- priority preemptive scheduling. No process in queue 2 can run unless queue 1 is empty. If there are no processes in queue 2, those in queue 3 will be elevated to queue 2. If a new process enters queue 1, while a process from queue 2 was running, that process will be preempted and new process from queue 1 will be given the control of CPU.

- **Shortest Remaining Time Next (SRTN) Scheduling with Priority**:

This scheduling algorithm is preemptive version of Shortest Job First (SJF) algorithm. The next process to be removed from the ready queue is the one with the shortest CPU burst time. If two or more processes are having the smallest burst time, the one with the higher priority is selected.

When a new process arrives at the ready queue, the scheduler will compare the remaining CPU time required for the currently running process with the next CPU burst time required for the newly arrived process.

If the next CPU burst time required for the newly arrived process is shorter, it will preempt the currently running process and it will be placed in the ready queue and newly arrived process will be given the control of CPU. But if the remaining time required for the currently running process is shorter, it will be allowed to continue.

Expected output:

1. User should be able to enter the details about the processes such as Arrival Time, Burst Time, Priority, Time Quantum assigned at the beginning of simulation and the number of processes can range from **3 to 10**.

2. Executing the program should show the **Gantt chart** (text form) of each algorithm.

3. Sample inputs and outputs:

Process	Burst time	Arrival time	Priority
P0	6	0	3
P1	4	1	3
P2	6	5	1
P3	6	6	1
P4	6	7	5
P5	6	8	6

Q1: FCFS with priority

P0	P2	P3	P0	P1	P4	P5	
0	5	11	17	18	22	28	34

Q2: RR with priority (quantum = 2)

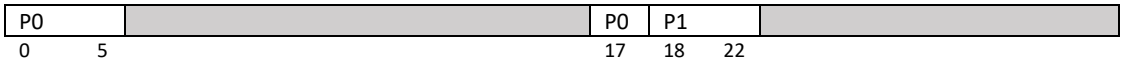
P0	P1	P0	P2	P3	P2	P3	P2	P3	P1	P0	P4	P5	
0	2	4	6	8	10	12	14	16	18	20	24	30	34

Q3: 3 level queue (quantum = 2)

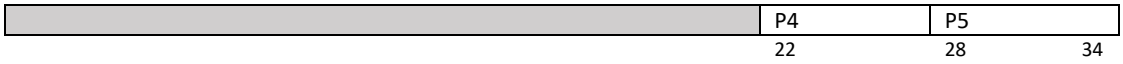
Queue 1

	P2	P3	P2	P3	P2	P3	
	5	7	9	11	13	15	17

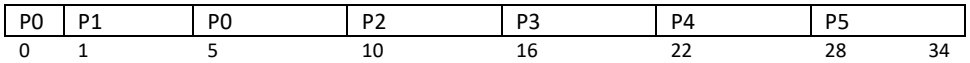
Queue 2



Queue 3



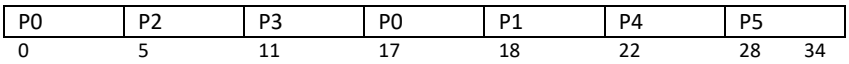
Q4: SRTN



Note: After P0 has finished, the order of the subsequent processes do not matter as they all have the same burst time.

4. You do not need to have any UI/UX. You can just represent the Gantt chart in text.

   Example Gantt Chart:



   In the shell/command line:

   0 P0 5 P2 11 P3 17 P0 18 P1 22 P4 28 P5 34

References:

- Abraham Silberschatz, Peter B. Galvin, & Greg Gagne, "Operating Systems Concepts", 9th Edition, John Wiley, 2014
- William Stallings, "Operating Systems-Internals and Design Principles", 7th Edition, Pearson, 2012
- David Petrou, John W. Milford, Garth A. Gibson, "Implementing Lottery Scheduling: Matching the Specializations in Traditional Schedulers", *Proceedings of the 1999 USENIX Annual Technical Conference, USA,* June 6-11, 1999.
- Carl A. Waldspurger, William E. Weihl, "Lottery Scheduling: Flexible Proportional-Share Resource Management" *Proceedings of the First Symposium on Operating System Design and Implementation*, November 1994.