San José State University
Department of Computer Engineering

CMPE 142
Operating Systems
Section 1

Spring 2021
Instructor: Ron Mak

Assignment #3

Assigned: Friday, February 12
Due: Friday, February 26 at 11:30 AM
Team assignment, 100 points max

Process scheduling algorithms

This assignment will give you experience with process scheduling algorithms. Write a C or C++ program that performs runs of the following process scheduling algorithms:

1. First-come first-served (**FCFS**) [nonpreemptive]
2. Shortest job first (**SJF**) [nonpreemptive]
3. Shortest remaining time (**SRT**) [preemptive]
4. Round robin (**RR**) [preemptive]
5. Highest priority first (**HPF**) [nonpreemptive] with 4 priority levels
6. Highest priority first (**HPF**) [preemptive] with 4 priority levels

Run each scheduling algorithm for 100 quanta (time slices), labeled 0 through 99. Before each run of an algorithm, create 20 **simulated processes**. Each simulated process is simply a small data structure that stores information about the process that it represents.

For each simulated process, randomly generate:

- An **arrival time**: a float value from quanta 0 through 99.
- An **expected total run time**: a float value from 0.1 through 10 quanta.
- A **priority**: integer 1, 2, 3, or 4 (1 is highest)
- Include any other attributes that you may need.

Tip: While debugging your program, you may want the same pseudo-random numbers each time. For this to happen, you should set the seed of the random number generator to a value, such as 0. Read about the `rand()` and `srand()` functions for C and C++:

- https://www.geeksforgeeks.org/rand-and-srand-in-ccpp/
- https://www.geeksforgeeks.org/generating-random-number-range-c/

Assume only one CPU and one ready queue. Sort the simulated processes in the ready queue by arrival time. Your process scheduler can do process switching only at the start of each time quantum. For this assignment, only consider CPU time for each process (no I/O wait times, no process switching overhead).

For RR, use a time slice of 1 quantum.

For HPF, use 4 priority queues. For preemptive scheduling, use RR with a time slice of 1 quantum for each priority queue. For non-preemptive scheduling, use FCFS. For both preemptive and non-preemptive schedule, add **aging** to help prevent starvation. After a process has waited for 5 quanta at a priority level, bump it up to the next higher level.

Each simulation run should last until the completion of the last process, even if it goes beyond 100 quanta. No process should get the CPU for the first time after time quantum 99.

Run each algorithm 5 times to get averages for the statistics below. Before each run, clear the process queue and create a new set of simulated processes.

Outputs for each algorithm run (total 30 runs)

Your output should include:

- The sorted **contents of the ready queue** before the start of the run.
    - Each created process's name (such as A, B, C, ...), arrival time, expected run time, and priority.
- A **timeline** of the 100+ quanta that shows which process ran during each quantum, such as ABCDABCD ...
    - Show a process's name in a quantum even if it completed execution before the end of that quantum.
    - The CPU can be idle during the last part of a quantum if a process completes before the end of the quantum. (No need to show the idle time within a quantum.)
    - Show a hyphen - if a quantum is completely unused.
- **Calculated statistics for each algorithm run**:
    - Average turnaround time for the processes that ran.
    - Average waiting time for the processes that ran.
    - How many processes in the ready queue that never ran.
    - Throughput of the algorithm per 100 quanta:

$$\frac{number\ of\ completed\ processes \times 100\ quanta}{total\ quanta\ to\ complete\ the\ processes}$$

Report

1 to 2 pages.

The average statistics over 5 runs for each scheduling algorithm.

Discuss which algorithm appears to be best for each of the calculated statistics.

What to submit

Submit a zip file to Canvas, **Assignment #3: Process Scheduling Algorithms,** that contains:

- Your C or C++ source files.
- A text file containing output from your simulation runs.
- Your report as a PDF.

Rubric

Your submission will be graded according to these criteria:

Criteria	Max points
Algorithm runs	**70**
• Ready queue contents	• 10
• Timeline of 100+ quanta	• 20
• Average turnaround time	• 10
• Average waiting time	• 10
• How many processes never ran	• 10
• Throughput	• 10
Report	**30**
• Average statistics for each algorithm over 5 runs.	• 20
• Discussion of which algorithm appears to be the best.	• 10