Lab 5 // Algorithms and Data Structures (CS210)

Dr. Pierpaolo Dondio

This lab contains exercise on CPU Scheduling techniques (mainly pen and paper exercises) and a java task using stacks.

CPU SCHEDULING ALGORITHMS

Review the CPU scheduling solutions shown in class, and try to understand the FCFS, SJF, (non-pre-emptive and pre-emptive, also known as SRTF – shot remaining time first), Round Robin (RR), and priority-based (pre-emptive and non-pre-emptive).

Remember the performance criteria used:

- **Turnaround time** = completion time − arrival time (time to execute the process).
- **Waiting time** = turnaround time – CPU burst time (time the process waits in the queue) .
- **Response time** = first scheduled time − arrival time (time to start executing the process)

We use the average of the above metrics to have an idea of the efficiency of the CPU scheduling algorithm.

Exercise 1: FCFS vs SJF

Given the following processes:

Process	Arrival Time	Burst Time
P1	0	8
P2	1	4
P3	2	9
P4	3	5

1. Construct the Gantt chart using **First-Come, First-Served (FCFS)**.
2. Calculate the performance metrics.
3. Repeat using **Shortest Job First (non-preemptive)**.
4. Compare results. Which algorithm gives better average waiting time? Why?

Exercise 2: Preemptive vs Non-preemptive Scheduling

Given the following processes:

Process	Arrival Time	Burst Time
P1	0	7
P2	2	4
P3	4	1
P4	5	4

1. Schedule using **Non-preemptive SJF** and compute metrics.
2. Schedule using **Preemptive SJF (SRTF)** and compute metrics.
3. Which process benefits most from preemption? Why?

Exercise 3: Round Robin (RR)

Given the following processes and **time quantum = 3**:

Process	Arrival Time	Burst Time
P1	0	10
P2	0	4
P3	1	5

1. Draw the Gantt chart for Round Robin.
2. Calculate the performance metrics.
3. Compare the results with **FCFS** scheduling. What is the effect of time quantum?

Exercise 4: Priority Scheduling

Given the following processes (lower number = higher priority):

Process	Arrival Time	Burst Time	Priority
P1	0	5	2
P2	1	3	1
P3	2	8	4
P4	3	6	3

1. Schedule the processes using **Non-preemptive Priority** scheduling.
2. Compute the performance metrics.
3. Discuss what might happen if a process with very low priority keeps getting delayed (starvation). How can aging solve this problem?

Exercise 5: Comparison & Discussion

1. For each scheduling algorithm (FCFS, SJF, SJF pre-emptive, RR, Priority):
    - State one **advantage** and one **disadvantage**.
    - Give one **real-world example** where it might be applied.
2. Consider a system with many interactive users (short tasks). Which algorithm would you prefer? Why?
3. Consider a batch-processing system with long, CPU-heavy jobs. Which algorithm is most efficient?

Exercise 6

The file ***CPUSchedulingLab.java*** on Moodle implements the above CPU scheduling algorithms in JAVA. You can download it , run it and try to understand its functioning.

Verify your answer with the JAVA implementation.

Exercise 7 – JAVA Stacks

Using the built-in function *java.util.Stack*, create a program that accept a string , and check if the string is a palindrome.

Use the pop() and push() operations of the stack to create the string and reverse it.

HINT: a string is a palindrome if the reversed string is equal to the original string**.**