Cairo University
Faculty of Engineering
Computer Engineering Dept.
CMPN 303

OS Scheduler

Introduction

It is required to implement an OS scheduler using different scheduling algorithms. The work is divided into two modules:

- "Process Generator": generates the processes to be scheduled.
- "Scheduler": produces the schedules based on the chosen algorithm and demonstrates these schedules by visual graphs.

Process Generator Module [30% of the Grade]

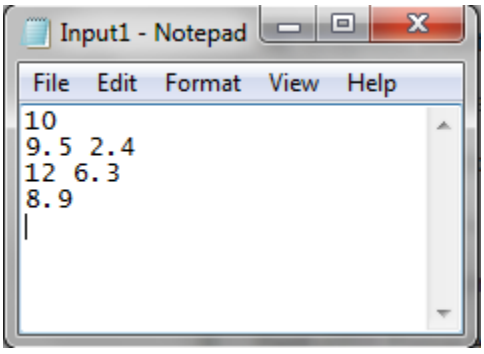
Each process has a set of parameters. Each parameter is generated randomly following a certain distribution as indicated:

1. Arrival Time : follows Normal distribution
2. Burst Time : follows Normal distribution
3. Priority : follows Poisson distribution

Input: is a text file organized as follows:

- First line should include the number of processes.
- Second line should include μ and σ of arrival time distribution separated by a white space.
- Third line should include μ and σ of burst time distribution separated by a white space.
- Fourth line should include λ of priority distribution.

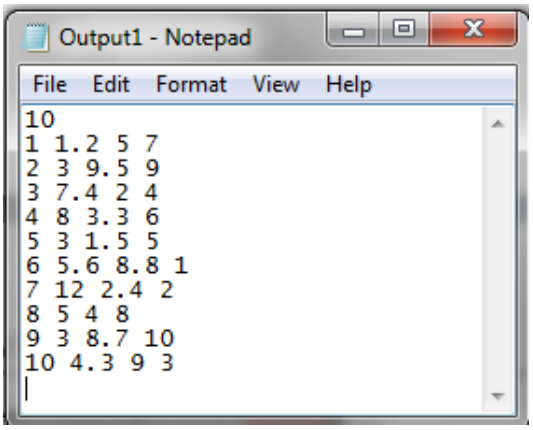
Input File Example:



Output: is a text file organized as follows:

- First line should include the number of processes.
- Each line contains the parameters for one process only, separated by a white space, in the following order: process number, arrival time, burst time and priority.

Output File Example: (N.B.: The values in the given file are imaginary ones)



Scheduler Module [70% of the Grade]

This module is responsible for generating a schedule for the current processes in the system to specify the CPU usage by these processes.

You are required to implement 3 scheduling algorithms:

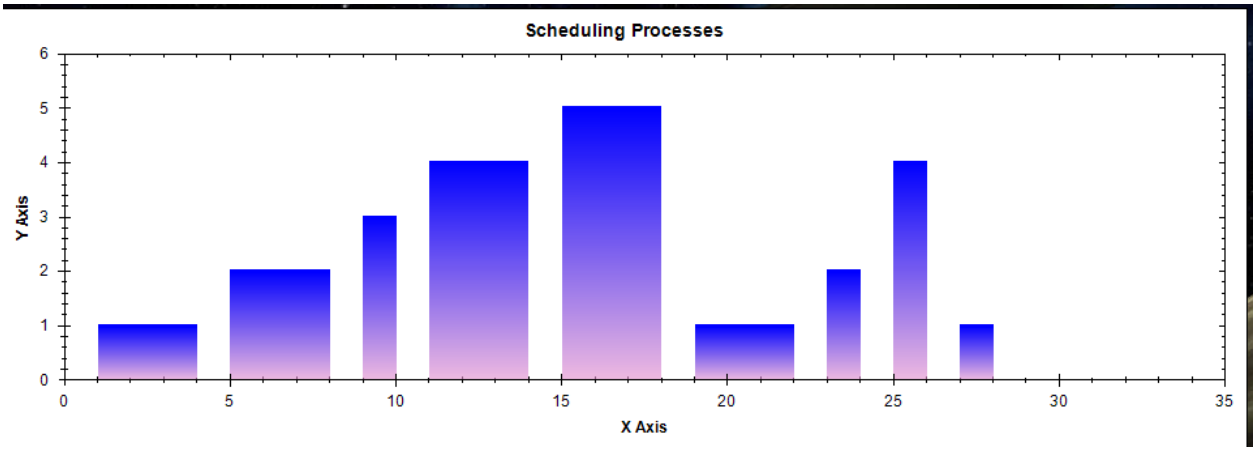
1. [15%] Non-Preemptive Highest Priority First. (HPF)
2. [15%] First Come First Served. (FCFS)
3. [20%] Round Robin with fixed time quantum. (RR)
4. [20%] Preemptive Shortest Remaining Time Next. (SRTN)

Input: Through a simple GUI the user should be able to do the following:

- Enter the input file name generated by the Process Generator Module.
- Choose one of the implemented scheduling algorithms to run.
- Specify the "Context Switching" time.
- Specify the "Time Quantum" in case of choosing Round Robin.

Output:

- A visual graph that shows the generated schedule.
  Output Graph Example:
  The x-axis represents the time while the y-axis represents the process number.



- A text file containing the following metrics for the generated schedule:
  - Waiting time of each process.
  - Turnaround time of each process.
  - Weighted Turnaround time of each process.
  - Average Turnaround time of the schedule.
  - Average Weighted turnaround time of the schedule.

Languages and Operating Systems

- You can use any programming language.
- You can work under any of the following OSs: Windows, Linux or MacOS.

General Guidelines

- The scheduler should be notified every time a new process has arrived in the system to act accordingly.
- Any tie should be broken by starting with the process whose number is smaller.
  For example: In HPF algorithm, if more than one process has the same priority, you should break the tie by starting with the process whose number is smaller.
- Assume that the memory size is infinite and that I/O requests are disabled.
- Assume that a greater number means a higher priority.
- To generate a graph, you can use any appropriate library such as: ZedGraph.
- In your calculations, do not ignore the context switching time.
- Stick to the file formats specified above, as you will be given testing scenarios.

Deliverables

- Samples of input/output text files generated by “Process Generator Module”. (at least 3 samples)
- Source Code.
- Executable file. (make sure that it runs correctly with no missing dependencies)

Work Load and Due Date

You should work in groups of 2 members. Delivery is on **Tuesday 27/11/2018**.