CPU Scheduler Simulator (20211208)

Objectives:

Your project is to write a program that simulates the service of jobs (process) by the CPU of a multi-user computer.

Inputs should be taken from text file with command arguments. And results also should be written to output file and screen.

Recommendations & Rules:

- The application **must be** written in C/C++ programming language and **must be** compiled on GNU/Linux Operating Systems.
- Don't use any special library for protocols when coding. Write your own functions if it's necessary. Only exemption is data structure functions that given in Data structures lecture.
- You should commit changes on **github** system. Usually at least one time a day you should commit your code on **github**.
- Unless otherwise explicitly specified, all written assignments or code that is submitted is to be entirely the student's own work. Using any code or copying any assignment from others is strictly prohibited without advance prior permission from the instructor.
- All students' work is their own. Students who do share their work with others are *as* responsible for academic dishonesty as the student receiving the material.
- Students who encourage others to cheat or plagiarize, or students who are aware of plagiarism or cheating and do not report it are also participating in academically dishonest behavior.
- In this case (academically dishonest behavior), students to be punished with no grade.

Project Details:

1. Program should have command argument options to get input data from a text file and command argument option for write output data. "-f" must be used for command argument and following text indicate input file name and "-o" for output file name.

```
student@db:~$ ./cpe351 –f input.txt –o output.txt
```

2. This file should have three columns and unlimited rows. It must be text and fields should delimit by ":" character. Below table represent structure of this file.

Burst Time	**Arrival Time**	**Priority**
10	0	0
5	0	0

Some data fields may not use at decision process so you can omits the values. But every process should have three of these values. At the Appendix section input.txt file given as example.

3. After first time executing simulator should ask for user to choose scheduling method. Menu should have at least four options.

```
CPU Scheduler Simulator
1) Scheduling Method (None)
2) Preemptive Mode (Off)
3) Show Result
4) End Program
Option >
```

   a) Scheduling Method: In this mode program optain scheduling method from user. You should implement:
      - None: None of scheduling method chosen
      - First Come, First Served Scheduling
      - Shortest-Job-First Scheduling
      - Priority Scheduling
      - Round-Robin Scheduling (You should also obtain time quantum value)

   b) Preemptive Mode: This is a clock-driven mode, in which the clock is a counter that is incremented each simulated second. The clock is initially 0, and continues "ticking" until all the jobs in the input have been serviced and left the system. This simulation runs until a particular number of jobs have been serviced.

Non-preemptive Mode: In this mode dispatcher should make decision after running process was terminated.

   c) Show Result: If user chooses this option, program shows report to user from screen (As described Step 4).

   d) End Program: If user chooses this option, program show all (implemented) scheduling results on display (As described Step 4) and also write these results to output file that given with “-o” option and terminate simulator.

   e) Appendix section has screen and file outputs. Please create same output for your program.

4. Your program must gather statistics, compute, and report the following information:
   a) The average waiting time that all jobs spend in the wait queue.
   b) The waiting time that each job spends in the wait queue.

5. You can get more information for CPU Scheduling methods from your text book (chapter 5). Some Scheduling methods not run in Preemptive mode. So take care about it.

APPENDIX

File input.txt sample content

```
5:0:3
4:1:2
3:1:1
4:2:2
3:3:1
```

output.txt/Screen output

```
Scheduling Method:  First Come First Served
Process Waiting Times:
P1: 0 ms
P2: 4 ms
P3: 8 ms
P4: 10 ms
P5: 13 ms
Average Waiting Time: 7 ms
```

```
Scheduling Method:  Shortest Job First – Non-Preemptive
Process Waiting Times:
P1: 0 ms
P2: 10 ms
P3: 4 ms
P4: 13 ms
P5: 5 ms
Average Waiting Time: 6.4 ms
```

```
Scheduling Method:  Shortest Job First –Preemptive
Process Waiting Times:
P1: 6 ms
P2: 10 ms
P3: 0 ms
P4: 13 ms
P5: 1 ms
Average Waiting Time: 6 ms
```

```
Scheduling Method:  Priority Scheduling  –Preemptive
Process Waiting Times:
P1: 14 ms
P2: 6 ms
P3: 0 ms
P4: 9 ms
P5: 1 ms
Average Waiting Time: 6 ms
```

Scheduling Method: Priority Scheduling – Non-Preemptive
Process Waiting Times:
P1: 0 ms
P2: 10 ms
P3: 4 ms
P4: 13 ms
P5: 5 ms
Average Waiting Time: 6.4 ms

Scheduling Method: Round Robin Scheduling – time_quantum=2
Process Waiting Times:
P1: 14 ms
P2: 9 ms
P3: 11 ms
P4: 11 ms
P5: 12 ms
Average Waiting Time: 11.4 ms