بسم الله الرحمن الرحيم

Processor Scheduling Algorithms

FIFO, SJF and RR

Objective:

All modern operating systems support multiprogramming and/or multithreading. This requires a very fast and efficient short-term (CPU) scheduling algorithm that responds to a dynamically changing workload, while maintaining system responsiveness and efficiency. In this lab, you are going to practice and test your understanding of how different schedulers perform under scheduling metrics such as response time, turnaround time, and total waiting time. The three basic scheduling algorithms used are: **FIFO**, **SJF**, and **RR**.

Background:

A program, written in *Python3*, called *scheduler.py*, is used in this lab to generate random CPU-scheduling problems. It works under any system that has *Python* interpreter version *3.8* or above. You can download *scheduler.py* from the labs page of the course web site. It can generate practice problems for all three basic CPU-scheduling policies you took in the class (**FIFO**, **SJF** and **Round-Robin**). When you run the program you can specify, on the command line, which policy to use, the number of jobs to be automatically generated for the run, and a random seed. Each time you change the random seed, a new problem is generated. You can also disregard the seed, and enter your own job list.

How to use the program:

1. Make sure you have *Python 3.8* or above installed in your system.
2. Download *scheduler.py* from the course web site, and note where you saved it.
3. Use the command interface and change to the directory you noted above.
4. At the prompt type:

```
> python ./scheduler.py [options]
```

The command-line options are:

```
-h,          --help               show this help message and exit
-s SEED,     --seed=SEED          the random seed – Default = 0
-j JOBS,     --jobs=JOBS          number of jobs in the system – Default = 3
-l JLIST,    --jlist=JLIST        instead of random jobs, provide a comma-separated
                                  list of run times
-m MAXLEN,   --maxlen=MAXLEN      max length of job – Default = 10
-p POLICY,   --policy=POLICY      sched policy to use: SJF, FIFO, RR – Default = FIFO
-q QUANTUM,  --quantum=QUANTUM    length of time slice for RR policy – Default = 1
```

For example: If you want the simulator to generate a problem to compute the response, turnaround, and waiting times for three jobs using a **FIFO** policy, then run the following command:

```
> python ./scheduler.py -p FIFO -j 3 -s 100
```

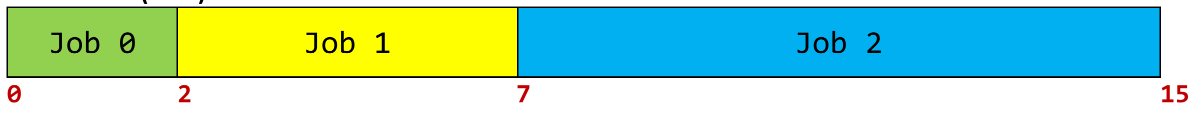
This specifies the **FIFO** policy with three jobs, and, importantly, a specific random seed of 100. If you later want to generate this exact same problem, you have to specify this exact same random seed again. The above command will generate for you the following problem, you should see:

```
ARG policy FIFO
ARG jobs 3
ARG maxlen 10
ARG seed 100

Here is the job list, with the run time of each job:
  Job 0 (length = 2)
  Job 1 (length = 5)
  Job 2 (length = 8)

Compute the turnaround time, response time, and wait time for each job.
You can use -s <somenumber> or your job list (-l 10,15,20 for example)
to generate different problems for yourself.
```

As you can see from this example, three jobs are generated: job 0 of length 2, job 1 of length 5, and job 2 of length 8.

5. As the program states, you can now solve the generated problem by forming a *Gantt* chart from the job order and their lengths, then computing some statistics, as follows:

Gantt chart (FIFO):



Final statistics:

```
 Job  0 -----> Response:  0.00      Turnaround    2.00      Wait  0.00
 Job  1 -----> Response:  2.00      Turnaround    7.00      Wait  2.00
 Job  2 -----> Response:  7.00      Turnaround  15.00       Wait  7.00

Average ---> Response:  3.00      Turnaround    8.00      Wait  3.00
```

6. Repeat as many times as required to fully understand all three algorithms.

Note:

For calculating the final statistics, we define the following:

- The ***Response time***: is the time a job spends waiting after arrival before first running,
- The ***Turnaround time***: is the time it took to complete the job since first arrival,
- The ***Wait time***: is the sum of all the time spent ready but not running.

If you want to try the same type of problem but with different inputs, try changing the number of jobs or the random seed or both. Different random seeds basically give you a way to generate an infinite number of different problems.

Exercise (4 points):

Each student shall generate and solve a different set of 4 problems, one of each policy type: **FIFO**, **SJF**, and two **RR**. Copy the generated problems, the parameters you used to generate each problem along with your solutions into one .pdf file and submit it to your TA.