Justin Cole - W1286374
Tejas Dedhiya - W1605246
Jianqiao Ge - W1609203
Piyush Kulkarni - W1629006
Shivani Deosatwar - W1588465

Advanced Operating System: COEN 383
Project 2: Process Scheduling Algorithms
Group No. 2

Introduction

Objective : This project gives us experience with process scheduling algorithms.

We have implemented Java programs that run the following process scheduling algorithms:

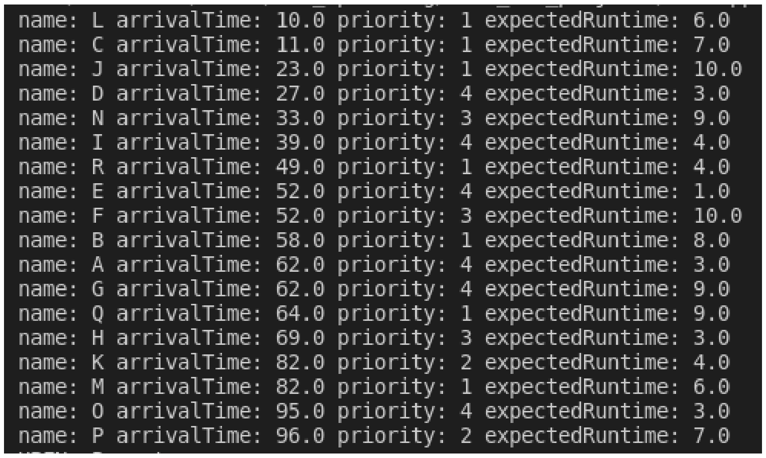
● First come first-served (FCFS) [non-preemptive]
● Shortest job first (SJF) [non-preemptive]
● Shortest remaining time (SRT) [preemptive]
● Round robin (RR) [preemptive]
● Highest priority first (HPF) [both non-preemptive and preemptive]

Assumptions: Each process has a random arrival time, a priority and expected run time. The algorithm will be run for 100 quanta. After that it will finish any processes that are partially complete, but start no new ones. There exists only one process queue. There is no I/O time. The highest priority first uses 4 queues. Round Robin uses 1 quanta time slice. Throughput is the amount of processes completed during the entire test. In addition, when a job is completed under HPF with aging, it is counted towards its original queue's throughput.
The below data represents the statistics for each algorithm.

Github: https://github.com/Jcole33/coen383_project2

Tests

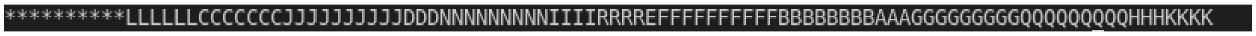
We generated 5 random workloads and selected the workload that seemed like a good average case.
The processes in that workload are:



```
name: L arrivalTime: 10.0 priority: 1 expectedRuntime: 6.0
name: C arrivalTime: 11.0 priority: 1 expectedRuntime: 7.0
name: J arrivalTime: 23.0 priority: 1 expectedRuntime: 10.0
name: D arrivalTime: 27.0 priority: 4 expectedRuntime: 3.0
name: N arrivalTime: 33.0 priority: 3 expectedRuntime: 9.0
name: I arrivalTime: 39.0 priority: 4 expectedRuntime: 4.0
name: R arrivalTime: 49.0 priority: 1 expectedRuntime: 4.0
name: E arrivalTime: 52.0 priority: 4 expectedRuntime: 1.0
name: F arrivalTime: 52.0 priority: 3 expectedRuntime: 10.0
name: B arrivalTime: 58.0 priority: 1 expectedRuntime: 8.0
name: A arrivalTime: 62.0 priority: 4 expectedRuntime: 3.0
name: G arrivalTime: 62.0 priority: 4 expectedRuntime: 9.0
name: Q arrivalTime: 64.0 priority: 1 expectedRuntime: 9.0
name: H arrivalTime: 69.0 priority: 3 expectedRuntime: 3.0
name: K arrivalTime: 82.0 priority: 2 expectedRuntime: 4.0
name: M arrivalTime: 82.0 priority: 1 expectedRuntime: 6.0
name: O arrivalTime: 95.0 priority: 4 expectedRuntime: 3.0
name: P arrivalTime: 96.0 priority: 2 expectedRuntime: 7.0
```

The results of each algorithm on that workload are as follows:

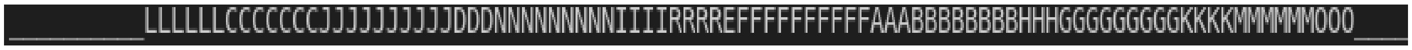
I. First-Come First-Serve (FCFS) Non-Preemptive



```
**********LLLLLLCCCCCCCJJJJJJJJJJDDDNNNNNNNNNIIIIRRRREFFFFFFFFFFBBBBBBBBAAAGGGGGGGGGQQQQQQQQQHHHKKKK
```

Average Response Time: 7.3333335
Average Waiting Time: 7.3333335
Average Turnaround Time: 13.333333
Throughput: 15

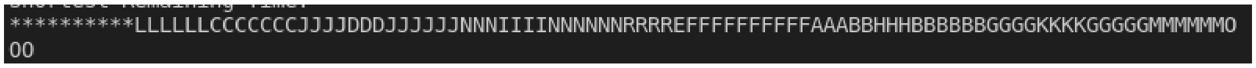
II. Shortest Job First (SJF) Non-Preemptive



```
__________LLLLLLCCCCCCCJJJJJJJJJJDDDNNNNNNNNNIIIIRRRREFFFFFFFFFFAAABBBBBBBBHHHGGGGGGGGGKKKKMMMMMMOOO___
```

Average Response Time: 4.5
Average Waiting Time: 4.5
Average Turnaround Time: 10.125
Throughput: 16.0

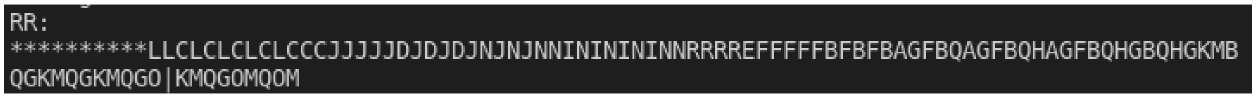
III. Shortest Remaining Time (SRT) Preemptive



```
**********LLLLLLCCCCCCCJJJJDDDJJJJJJNNNIIIINNNNNNRRRREFFFFFFFFFFAAABBHHHBBBBBBGGGGKKKKGGGGGMMMMMMO
OO
```

Average Response Time: 0.5833333
Average Waiting Time: 0.75
Average Turnaround Time: 1.8214285
Throughput: 16.0

IV. <u>Round-Robin (RR) Preemptive</u>



```
RR:
**********LLCLCLCLCLCCCJJJJJDJDJDJNJNJNNINININININNRRRREFFFFFBFBFBAGFBQAGFBQHAGFBQHGBQHGKMB
QGKMQGKMQGO|KMQGOMQOM
```

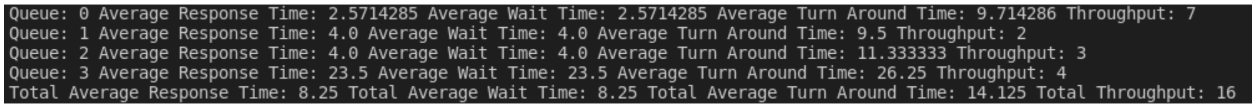
Average Response Time: 2.0588236

Average Wait Time: 12.058824

Average Turnaround Time: 17.882353

Throughput: 17

V. <u>Highest Priority First (HPF) Non-Preemptive</u>



```
Queue: 0 Average Response Time: 2.5714285 Average Wait Time: 2.5714285 Average Turn Around Time: 9.714286 Throughput: 7
Queue: 1 Average Response Time: 4.0 Average Wait Time: 4.0 Average Turn Around Time: 9.5 Throughput: 2
Queue: 2 Average Response Time: 4.0 Average Wait Time: 4.0 Average Turn Around Time: 11.333333 Throughput: 3
Queue: 3 Average Response Time: 23.5 Average Wait Time: 23.5 Average Turn Around Time: 26.25 Throughput: 4
Total Average Response Time: 8.25 Total Average Wait Time: 8.25 Total Average Turn Around Time: 14.125 Total Throughput: 16
```

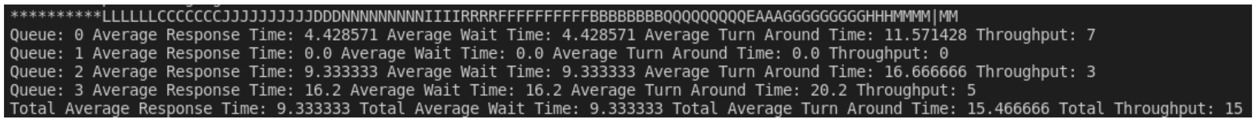
Average Response Time: 8.25

Average Wait Time: 2.8125

Average Turnaround Time: 14.125

Throughput: 16

VI. <u>Highest Priority First (HPF) Non-Preemptive with Aging</u>



```
**********LLLLLLCCCCCCCJJJJJJJJJJDDDNNNNNNNNNIIIIRRRRFFFFFFFFFFBBBBBBBBQQQQQQQQEAAAGGGGGGGGGHHHMMMM|MM
Queue: 0 Average Response Time: 4.428571 Average Wait Time: 4.428571 Average Turn Around Time: 11.571428 Throughput: 7
Queue: 1 Average Response Time: 0.0 Average Wait Time: 0.0 Average Turn Around Time: 0.0 Throughput: 0
Queue: 2 Average Response Time: 9.333333 Average Wait Time: 9.333333 Average Turn Around Time: 16.666666 Throughput: 3
Queue: 3 Average Response Time: 16.2 Average Wait Time: 16.2 Average Turn Around Time: 20.2 Throughput: 5
Total Average Response Time: 9.333333 Total Average Wait Time: 9.333333 Total Average Turn Around Time: 15.466666 Total Throughput: 15
```

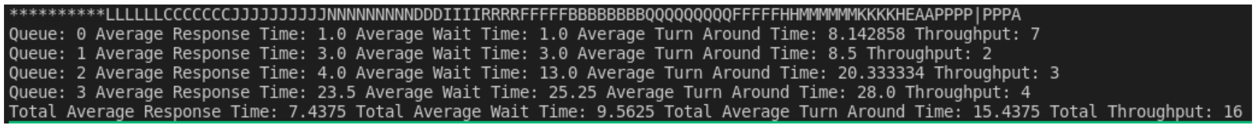
Average Response Time: 9.333333

Average Wait Time: 9.333333

Average Turnaround Time: 15.466666

Throughput: 15

VII. <u>Highest Priority First (HPF) Preemptive</u>



```
**********LLLLLLCCCCCCCJJJJJJJJJJNNNNNNNNNDDDIIIIRRRRFFFFFBBBBBBBBQQQQQQQQFFFFFHHMMMMMKKKKHEAAPPPP|PPPA
Queue: 0 Average Response Time: 1.0 Average Wait Time: 1.0 Average Turn Around Time: 8.142858 Throughput: 7
Queue: 1 Average Response Time: 3.0 Average Wait Time: 3.0 Average Turn Around Time: 8.5 Throughput: 2
Queue: 2 Average Response Time: 4.0 Average Wait Time: 13.0 Average Turn Around Time: 20.333334 Throughput: 3
Queue: 3 Average Response Time: 23.5 Average Wait Time: 25.25 Average Turn Around Time: 28.0 Throughput: 4
Total Average Response Time: 7.4375 Total Average Wait Time: 9.5625 Total Average Turn Around Time: 15.4375 Total Throughput: 16
```

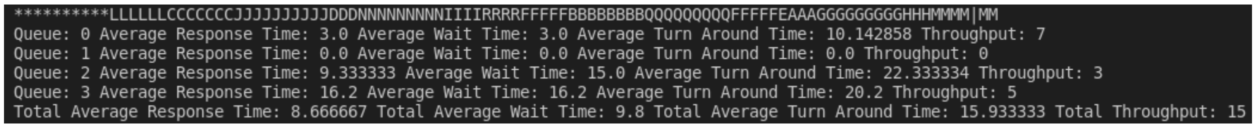
Average Response Time: 7.4375

Average Wait Time: 9.5625

Average Turnaround Time: 15.4375

Throughput: 16

VIII. Highest Priority First (HPF) Preemptive with Aging



```
**********LLLLLLCCCCCCCJJJJJJJJJDDDNNNNNNNNNIIIIRRRRFFFFFBBBBBBBBQQQQQQQQFFFFFEAAAGGGGGGGGGHHHMMMM|MM
Queue: 0 Average Response Time: 3.0 Average Wait Time: 3.0 Average Turn Around Time: 10.142858 Throughput: 7
Queue: 1 Average Response Time: 0.0 Average Wait Time: 0.0 Average Turn Around Time: 0.0 Throughput: 0
Queue: 2 Average Response Time: 9.333333 Average Wait Time: 15.0 Average Turn Around Time: 22.333334 Throughput: 3
Queue: 3 Average Response Time: 16.2 Average Wait Time: 16.2 Average Turn Around Time: 20.2 Throughput: 5
Total Average Response Time: 8.666667 Total Average Wait Time: 9.8 Total Average Turn Around Time: 15.933333 Total Throughput: 15
```

Average Response Time: 8.666667
Average Wait Time: 9.8
Average Turnaround Time: 15.933333
Throughput: 15

Observations

I. FCFS - This algorithm seems to have one of the worst throughputs and while it doesn't have the worst turnaround and wait time, it also isn't one of the best.

II. SJF - This algorithm has one of the better response, waiting, and turnaround times while still maintaining a decent throughput. The only major issue with this algorithm is that certain jobs may be starved if shorter jobs keep arriving before it gets a chance at the CPU.

III. SRT - This algorithm has the best response, waiting, and turnaround times by far. It also maintains a respectable throughput. However, this algorithm exacerbates the starvation problem that SJF had. Now long jobs not only have to hope that short jobs don't appear before they start running, but also that they don't appear while the longer job is running. This makes long jobs receive even less cpu time than they already had under SJF. This means that while its numbers in this test look fantastic, they are likely skewed by the limited subset of jobs that are actually polled.

IV. RR- Round Robin has a decent response time since every process is hit pretty quickly. However, its wait and turnaround times are abysmal. Everything has to wait for every other job to get a turn, meaning that even jobs that only need one or two quanta have to wait a lot more time than they would under other algorithms.

V. HPF Non-Preemption - Under HPF without preemption the experience largely depends on which priority the job sits at. On the higher priority levels response time is amazing. However, as one goes down the tiers it gets worse and worse until it reaches 23.5 quanta on the lowest end. This algorithm is better if you really want certain processes to have a fast response, at the cost of slowing all other processes down. In addition, starvation is a problem on the lower levels as well.

VI. HPF Non-Preemption with Aging - The statistics for our run of HPF without preemption, but with aging are interesting. As one would think that adding aging would allow for processes to be seen quicker and as such decrease wait times. However, because of the nature of our test, the aging process also exposes some previously starved processes in regular HPF without preemption. This makes the overall statistics for this run look worse even though it makes the statistics for the lower tiers better.

VII. HPF Preemption - HPF preemptive seems to stretch out the extremes of the statistics. The first queue has better statistics than when it was under the Non-Preemption policy, but the last queue also has worse statistics. This is similar to the difference between SJF and SRT. Jobs on lower tiers have to hope that no jobs arrive before or during their execution in order to get CPU time. As such starvation is even a bigger issue here.

VIII. HPF Preemption with Aging - The aging here has a similar effect as it did with the HPF Non-Preemption algorithm. The negative effect on the lower priority processes has been reduced at the cost of overall worse statistics for the algorithm. That being said however, the statistics for the higher priority processes look better than they did under HPF Non-Preemption with aging. This might mean that adding both Preemption and Aging might help to limit both of their worse side effects.

Conclusion

Overall we believe that there are a few processes that warrant interest. The SRT algorithm certainly presents lightning quick statistics, however, the fear of starvation for larger processes holds us back from selecting it. This makes the algorithms that incorporate aging very attractive. In addition, allowing for the speeding up of important processes gives the algorithm more flexibility to work in environments with various kinds of workloads. Finally, the Preemption algorithms seem to overall have better statistics during our tests. While there is the increased danger of starvation, we believe that the aging algorithm should offset that to acceptable levels. Therefore the algorithm that we believe is the best is the HPF Preemption algorithm that utilizes Aging.