COIS 3320 Lab 2

### Analysis

Results in my Log file as well.

| Job 1 | Average Turnaround Time | Average  Response Time | Number of context Switches | Start Time | End Time |
| --- | --- | --- | --- | --- | --- |
| FCFS | 187403 | 187254 | 0 | 88 | 745237 |
| SJF | 126858 | 126773 | 0 | 88 | 426121 |
| STCF | 198074 | 197910 | 4711 | 88 | 817888 |
| Round Robin (quantumSize = 30) | 203865 | 62681 | 15855 | 88 | 475738 |
| Round Robin (quantumSize = 75) | 179977 | 84168 | 6652 | 88 | 498988 |

| Job 2 | Average Turnaround Time | Average  Response Time | Number of context Switches | Start Time | End Time |
| --- | --- | --- | --- | --- | --- |
| FCFS | 39717 | 39628 | 0 | 88 | 446808 |
| SJF | 7661 | 7641 | 0 | 88 | 103459 |
| STCF | 27831 | 27694 | 4966 | 88 | 516568 |
| Round Robin (quantumSize = 30) | 58296 | 10211 | 6435 | 88 | 193138 |
| Round Robin (quantumSize = 75) | 151435 | 66424 | 5916 | 88 | 443788 |

| Job 3 | Average Turnaround Time | Average  Response Time | Number of context Switches | Start Time | End Time |
| --- | --- | --- | --- | --- | --- |
| FCFS | 338579 | 338368 | 0 | 88 | 1052746 |
| SJF | 14551 | 14528 | 0 | 88 | 116643 |
| STCF | 307068 | 306834 | 5329 | 88 | 1125298 |
| Round Robin (quantumSize = 30) | 197395 | 74969 | 18321 | 88 | 549718 |
| Round Robin (quantumSize = 75) | 375257 | 187424 | 11549 | 88 | 866263 |

### Summary

Job 1:

For job 1 my Global Process Array was populated with 5000 Process objects, that represent medium length jobs. According to the results, the Shortest Job First had the best turnaround time, meaning on average it finished executing a job faster than other jobs. This could be because the shortest jobs were completed first, and even though it was a non-preemptive algorithm like the FCFS it still performed better in all the other metrics. Meaning the FCFS was not very efficient.

As for my preemptive algorithms, the Round Robin with a quantum size of 75 had the best response time and fewest context switches, while the Round Robin with a quantum size of 30 had the best turnaround time. As for my STCF (preemptive shortest job first) it had the worst performance in both Average Turnaround Time, Average Response Time and context switches.

The reason why my RR timeslice of 30 had a better turnaround time than my RR with a time slice of 75 was due to the fact that shorter jobs were completed quicker since we switched between processes more frequently.

Though I am skeptical about my results for the STCF, I believe the reason for this was because of the nature of the Jobs and the preemptive nature of the algorithm. Which made it a poor choice for running medium jobs. The pre-empt time of the STCF also had a time of 30 which was also a factor in it, compared to my Round Robin with CPU timeslice of 75 that had more time to run jobs and did not prioritize jobs based on their time left.

Job 2

For job 2 my Global Process Array was populated with 5000 Process objects, which populated with a long jo to short job split of 80-20, meaning the majority of the jobs were short. Again my shortest job to completion job performed the best out of all 4 and the FCFS performed as expected. It had the worst response time and was the worst of the two non-preemptive algorithms since the shorter jobs were delayed by the longer jobs.

As for my preemptive algorithms, the Round Robin with a quantum size of 30 had the best Average Response Time and most context switches. While the STCF had satisfying results compared to my job 1, it had the shortest turnaround time and the fewest context switches. the Round Robin with a quantum size of 75 had the worst performance out of all 3 preemptive algorithms. It is also important to note the STCF took the longest to complete out of all 3.

The reason for this could be because the RR with a quantum size of 30 frequently did process switches and allowed the shorter jobs to finish quicker. While the RR with a quantum size of 75 did fewer context switches hence shorter jobs took longer than they needed, to complete.

Job 3

For job 3 my Global Process Array was populated with 5000 Process objects, which represented a long-short job split of 20-80. Again my SJF had the best Average response and turnaround time out of all 5 algorithms and also the shortest time. This could be due to it scheduling the quickest jobs first and no context switches.

As for my preemptive algorithms, the round robin with a quantum size of 30 had the best Average response and turnaround time, the round robin with a quantum size of 75 had the best Average Response Time but the worst Average turnaround Time and the STCF took the longest to complete.

The reason for this could be the shorter time slice allowed for more frequent context switching in the RR with the timeslice of 30, which may have helped to complete shorter jobs faster and improve response and turnaround times. The STCF took the longest to complete because the number of shorter jobs in the array was smaller compared to the loner jobs. While the RR with timeslice of 75 had a good turnaround time but a bad response time, this could be because the longer time slice allowed for longer-running jobs to complete faster, but it may have also led to delays for shorter jobs.

### General Analysis

According to my results from all 3 jobs the SFJ performed fantastically in all 3 scenarios, however, it is important to remember that it is a non-preemptive algorithm so longer jobs might end up hugging up the CPU and might not be the best algorithm to go for in most cases. The Round Robin (quantumSize = 30) also performed better than Round Robin (quantumSize = 75) in most cases but had more context switches, hence more overhead. While the FCFS performed as expected in all cases and the STFC.

In terms of the timeslice for the preemptive algorithms, smaller timeslice resulted in more context switches and better response times. However, this comes at the cost of increased overhead due to the context switches. A larger timeslice size would result in fewer context switches and potentially better performance, but it comes at the cost of longer response times for shorter jobs.