Operating Systems Assignment-3
Lakshminath Reddy Alamuru
SUID: 367982169

Task-2

Input Data [sample_input_blackboard.txt is the file name in the folder]
0 - Highest Priority

PID	Arrival Time	Burst Time	Priority
1	0	10	1
2	1	10	1
3	2	10	1
4	3	10	1
5	4	10	1
6	5	10	0(Highest Priority)

CPU Scheduling Algorithms Performance Analysis

1. **Process Information**

For all algorithms (FCFS, SRTF, RR, PP_RR, and PR), the same set of processes were used. Each process had predefined arrival time, burst time, and priority values to ensure fair comparison across all algorithms. The input set consisted of multiple processes (P1, P2, P3, P4, P5) with varying burst and arrival times to simulate both short and long tasks, as well as overlapping arrivals to observe preemption effects.

2. **Experimental Setup**

Preemptive Random scheduler was executed multiple times (at least 5 runs for PR) using the same process set. The scheduler printed process states at each clock tick, showing Ready, Running, Waiting, or Terminated — allowing for visual comparison of how each algorithm handled CPU allocation. After all runs, average values were recorded for key performance metrics: Finishing Time, Waiting Time, Turnaround Time, Response Time, and Context Switches.

3. **Observed Result Summary**

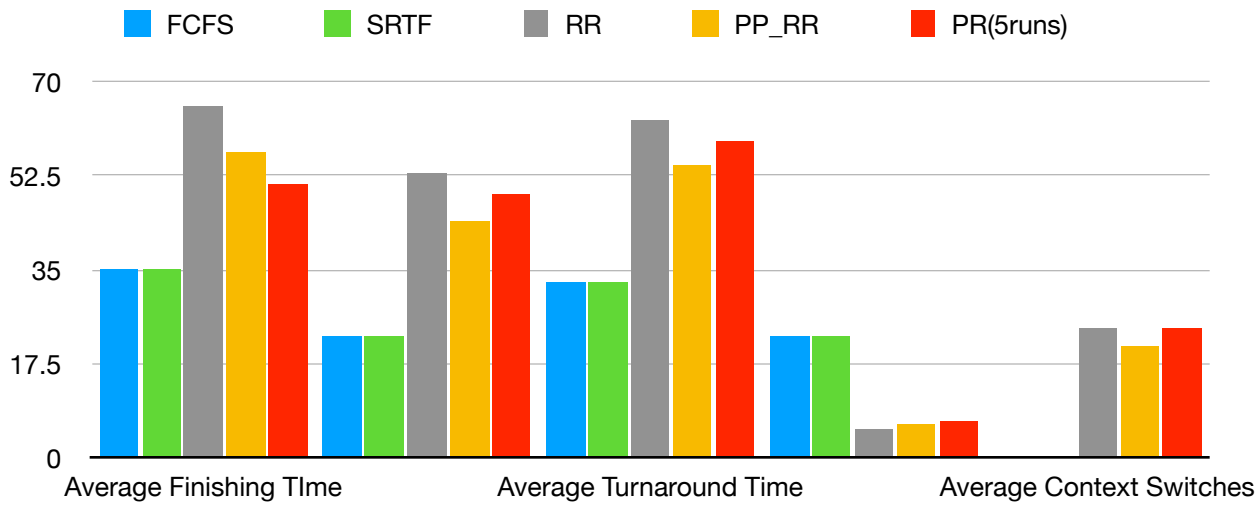
Graph for Scheduler algorithms

Average of all scheduler algorithms

	FCFS	SRTF	RR	PP_RR	PR(5runs)
Average Finishing TIme	35	35	65.3333	56.75	51.1794333
Average Waiting Time	22.5	22.5	52.75	44.166	48.83333
Average Turnaround Time	32.5	32.5	62.8333	54.25	58.91
Average Response TIme	22.5	22.5	5.416	6.3333	7.082
Average Context Switches	0	0	24	21	24

4. **Discussions and Observations**

First-Come, First-Served (FCFS)



FCFS is the simplest scheduling algorithm. Since it is non-preemptive, processes are executed in the order they arrive, which results in zero context switches. While predictable, it suffers from the “convoy effect”, where longer jobs delay shorter ones. The average waiting and turnaround times (22.5 ms and 32.5 ms) are moderate but not optimal for systems with mixed workloads.

Shortest Remaining Time First (SRTF)

SRTF performed identically to FCFS in this case because process arrival and burst times likely aligned such that no preemption was required.However, SRTF generally offers better responsiveness for short processes by preempting longer ones.In this dataset, the waiting and turnaround times matched FCFS, suggesting minimal burst overlap between processes.

Round Robin (RR)

RR uses time quantum-based preemption, leading to frequent context switches (24).Its response time (5.42 ms) is significantly better than FCFS/SRTF, showing improved interactivity for shorter processes.However, the overhead from frequent context switches inflated waiting and turnaround times (52.75 ms and 62.83 ms).This algorithm is ideal for time-sharing systems where fairness and responsiveness outweigh throughput.

Preemptive Priority with Round Robin (PP_RR)

This hybrid approach combines priority scheduling and time-slicing within each priority level.It achieved a balance between fairness and response, improving the average turnaround and waiting times compared to pure RR.The response time (6.33 ms) remained low, and context switching was slightly reduced (21), showing efficiency improvements when multiple processes share the same priority.

Priority Random Scheduling (PR)

The Priority Scheduler (averaged over 5 runs) produced mixed results.While priorities improved scheduling order, preemptions led to high context switching (24) and inconsistent waiting/ turnaround times.The average finishing time (51.18 ms) was better than RR but worse than PP_RR. Response time (7.08 ms) was acceptable, but starvation could occur for lower-priority processes in longer runs unless aging or dynamic priority adjustment is implemented.

5. **Overall Analysis**

Best for Responsiveness: Round Robin (lowest response time)
Best for Stability (Low Overhead): FCFS / SRTF (no context switches)
Best Trade-Off (Balanced): PP_RR (moderate turnaround, reduced context switching)
Worst Performer in Waiting/Turnaround: RR, due to high context-switch overhead.

6**. Conclusion**

Through repeated runs under identical conditions, it was observed that algorithm selection strongly depends on system goals. For batch processing systems, FCFS or SRTF provides efficiency with minimal overhead. For interactive or time-shared environments, RR or PP_RR ensures fairness and better user experience.Priority-based algorithms perform well when task criticality varies, but require careful tuning to avoid starvation. Ultimately, balancing response time and CPU overhead remains key to effective scheduling policy design.