PROCESS MANAGEMNENT

INTRODUCTION

- CPU and process management are critical for OS performance.
- This project simulates FCFS and Round Robin (RR) scheduling algorithms in Java.
- It generates random processes and displays key metrics like process IDs, arrival times, burst times, completion times, and waiting times.

PROBLEM STATEMENT

- Managing CPU time for multiple processes is challenging.
- Key issues include fair resource allocation, minimizing waiting times, and handling varying burst times.
- Poor scheduling can cause starvation and inefficiency.

PROPOSED SOLUTION

- FCFS: Processes are scheduled in the order they arrive. Simple but can cause high waiting times.
- RR: Processes get equal time slices (quantum). Ensures fairness, especially for short tasks.
- User Input: Choose the algorithm and generate processes.

WORKING

ava-based: Object-oriented design for modularity.

- Process Class: Stores details and metrics.
- Scheduler Class: Implements FCFS and RR.
- Display Module: Presents results clearly.

METHODOLOGY

- Process Generation: Random processes with unique IDs, arrival, and burst times.
- Input: User selects the algorithm and, for RR, the time quantum.
- Execution:
- FCFS sorts by arrival time.
- RR cycles through processes with time slices.
- Metrics: Completion, turnaround, and waiting times are calculated and displayed.
- Testing: Varying process numbers and quantum for accuracy.

RESULTS

- FCFS: Simple but can lead to long waits.
- RR: Ensures fairness and reduces wait times for shorter tasks.

KEY TAKEAWAYS

- FCFS: Efficient for certain tasks but can cause delays.
- RR: Fair and ideal for time-sharing systems.
- Educational Tool: Demonstrates core scheduling concepts.