Course T2Y2: Operating System
Lecturer: Heng Rathpisey

Name: Chheng Bunheang
ID: IDTB100048
Group: 1 SE
Gen10

Problem Set 1: CPU Scheduling

1. Introduction

This report outlines the development of a CPU scheduling simulator implementing four algorithms: First-Come, First-Served (FCFS), Shortest-Job-First (SJF), Shortest-Remaining-Time (SRT), and Round Robin (RR). The program allows users to select an algorithm, input process details, and view scheduling results, including the Gantt Chart, Waiting Time, Turnaround Time, and their averages. Error handling ensures valid user inputs.

2. Objective

The objective of this project is to implement a CPU scheduling simulator that:

- Supports **FCFS, SJF, SRT, and RR** algorithms.
- Accepts **user-defined processes** and parameters.
- Generates and displays **Gantt Charts, Waiting Times, Turnaround Times**, and their averages.
- Provides a **menu-driven interface** for user interaction.
- Implements **error handling** for invalid inputs.

3. Program Design and Implementation

3.1. Menu-Driven Interface

The program presents a menu allowing users to select a scheduling algorithm and input:

- Process ID (e.g., P1, P2, etc.)
- Arrival Time (when the process enters the queue)
- Burst Time (execution time required)
- Time Quantum (for RR only)

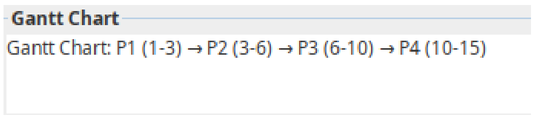
3.2. Algorithm Implementation

- FCFS: Processes execute in order of arrival. No preemption.
- SJF: Non-preemptive; the shortest burst time process executes first. Ties are resolved using FCFS.
- SRT: Preemptive SJF; processes with the shortest remaining burst time execute first.
- RR: Each process gets a fixed time quantum before preemption, ensuring fair CPU allocation.

3.3. Output Results

After the scheduling algorithms are executed, the following results are displayed:

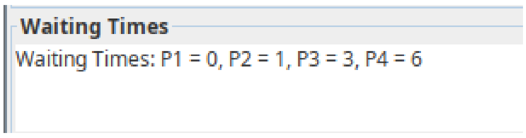
- ***Gantt Chart***: A visual timeline showing the execution order of processes.



Gantt Chart

Gantt Chart: P1 (1-3) → P2 (3-6) → P3 (6-10) → P4 (10-15)

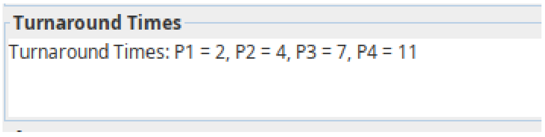
- ***Waiting Time***: Calculated as the total time a process has been waiting in the ready queue before its execution



Waiting Times

Waiting Times: P1 = 0, P2 = 1, P3 = 3, P4 = 6

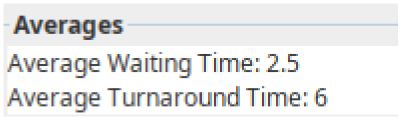
- ***Turnaround Time***: The total time a process spends from its arrival to its completion.



Turnaround Times

Turnaround Times: P1 = 2, P2 = 4, P3 = 7, P4 = 11

- ***Average Waiting Time***: The mean waiting time for all processes.
- ***Average Turnaround Time***: The mean turnaround time for all processes.



Averages

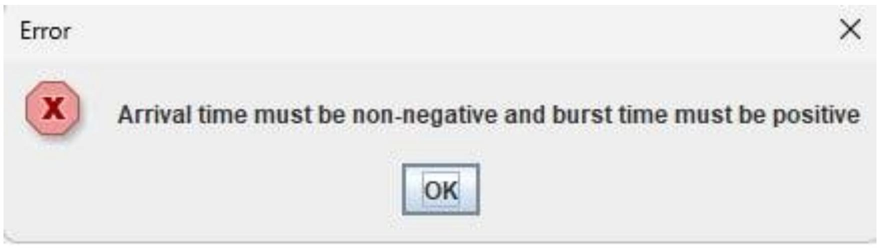
Average Waiting Time: 2.5

Average Turnaround Time: 6

<u>3.4. Error Handling</u>

The program is equipped with error handling mechanisms to handle invalid inputs. These include:

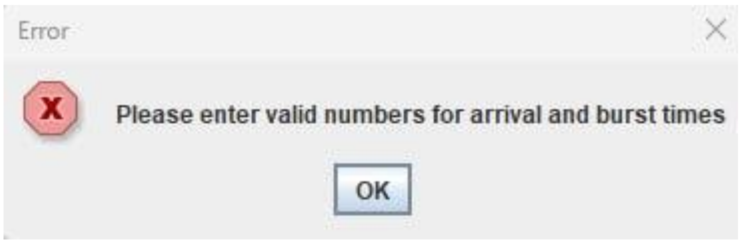
- ***Negative burst times***: An error message is displayed if a burst time is negative.



Error

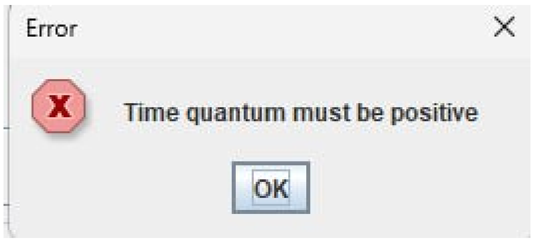
Arrival time must be non-negative and burst time must be positive

- ***Non-integer inputs***: The program checks if input values are integers, and if not, prompts the user to enter valid values.


Error

Please enter valid numbers for arrival and burst times

- ***Invalid time quantum in Round Robin***: If a time quantum is not positive, the program will ask for a valid input.


Error

Time quantum must be positive

3.5. Time Complexity

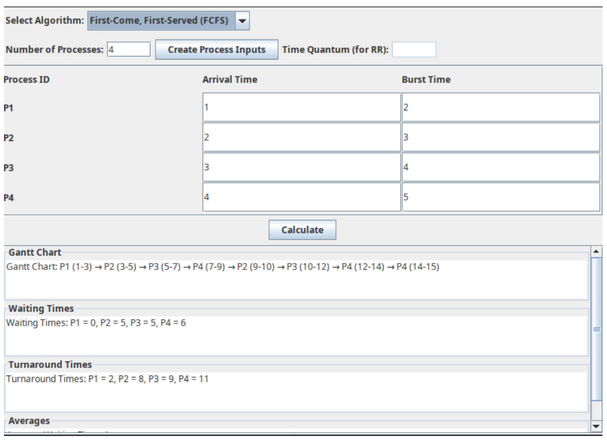
The time complexity of each scheduling algorithm is as follows:

- ***FCFS***: O(n), where n is the number of processes (as each process is checked once).
- ***SJF***: O(n^2), as sorting processes based on burst time is required.
- ***SRT***: O(n^2), as at each step, the algorithm checks all processes to find the shortest remaining time.
- ***Round Robin***: O(n * q), where n is the number of processes, and q is the number of time slices needed for each process to complete.

4. Results and Evaluation

The simulator was tested with various sets of input data for each scheduling algorithm. The results were evaluated based on the Gantt chart and the correctness of calculated waiting and turnaround times. The program was able to successfully generate the correct Gantt charts and compute the waiting and turnaround times for all scheduling algorithms.

Sample Output for FCFS:


Select Algorithm: First-Come, First-Served (FCFS)

Number of Processes: 4 Create Process Inputs Time Quantum (for RR):

Process ID	Arrival Time	Burst Time
P1	1	2
P2	2	3
P3	3	4
P4	4	5

Calculate

Gantt Chart

```
Gantt Chart: P1 (1-3) → P2 (3-5) → P3 (5-7) → P4 (7-9) → P2 (9-10) → P3 (10-12) → P4 (12-14) → P4 (14-15)
```

Waiting Times

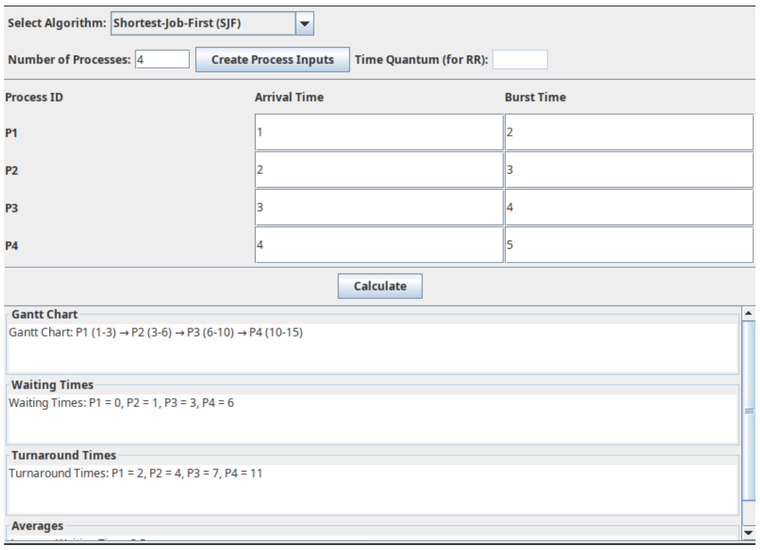
```
Waiting Times: P1 = 0, P2 = 5, P3 = 5, P4 = 6
```

Turnaround Times

```
Turnaround Times: P1 = 2, P2 = 8, P3 = 9, P4 = 11
```

Averages

Sample Output for SJF:


Select Algorithm: Shortest-Job-First (SJF)

Number of Processes: 4 Create Process Inputs Time Quantum (for RR):

Process ID	Arrival Time	Burst Time
P1	1	2
P2	2	3
P3	3	4
P4	4	5

Calculate

Gantt Chart

```
Gantt Chart: P1 (1-3) → P2 (3-6) → P3 (6-10) → P4 (10-15)
```

Waiting Times

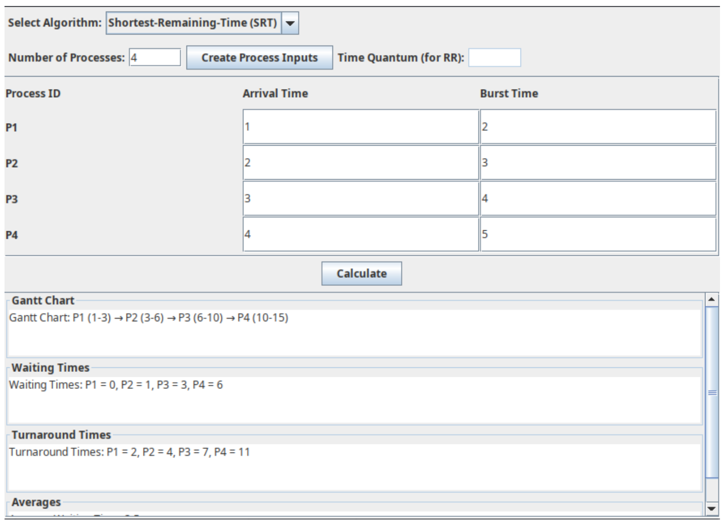
```
Waiting Times: P1 = 0, P2 = 1, P3 = 3, P4 = 6
```

Turnaround Times

```
Turnaround Times: P1 = 2, P2 = 4, P3 = 7, P4 = 11
```

Averages

***Sample Output for SRT*:**


Select Algorithm: Shortest-Remaining-Time (SRT)

Number of Processes: 4 Create Process Inputs Time Quantum (for RR):

Process ID	Arrival Time	Burst Time
P1	1	2
P2	2	3
P3	3	4
P4	4	5

Calculate

Gantt Chart

```
Gantt Chart: P1 (1-3) → P2 (3-6) → P3 (6-10) → P4 (10-15)
```

Waiting Times

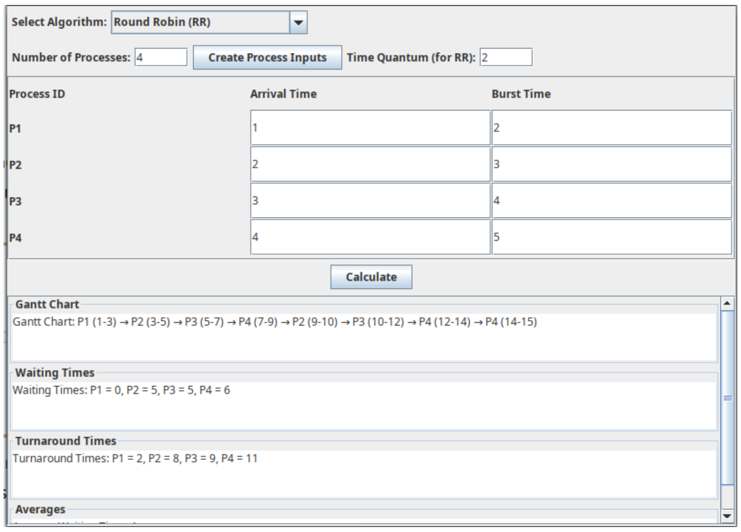
```
Waiting Times: P1 = 0, P2 = 1, P3 = 3, P4 = 6
```

Turnaround Times

```
Turnaround Times: P1 = 2, P2 = 4, P3 = 7, P4 = 11
```

Averages

***Sample Output for RR*:**



Select Algorithm: Round Robin (RR)

Number of Processes: 4 Time Quantum (for RR): 2

Process ID	Arrival Time	Burst Time
P1	1	2
P2	2	3
P3	3	4
P4	4	5

Gantt Chart

```
Gantt Chart: P1 (1-3) → P2 (3-5) → P3 (5-7) → P4 (7-9) → P2 (9-10) → P3 (10-12) → P4 (12-14) → P4 (14-15)
```

Waiting Times

```
Waiting Times: P1 = 0, P2 = 5, P3 = 5, P4 = 6
```

Turnaround Times

```
Turnaround Times: P1 = 2, P2 = 8, P3 = 9, P4 = 11
```

Averages

5. Conclusion

The CPU scheduling simulator successfully demonstrates the functionality of the four selected algorithms: FCFS, SJF, SRT, and Round Robin. The program correctly handles user input, generates Gantt charts, and calculates waiting and turnaround times. The error handling mechanisms ensure that invalid inputs do not crash the program and prompt the user to correct their mistakes.

This project provides a comprehensive understanding of CPU scheduling algorithms and their implementation, making it a valuable tool for students and developers working with process management.

6. Future Work

- ***Priority Scheduling****:* Future extensions could involve implementing priority-based scheduling.
- ***Preemptive SJF***: A more detailed version of preemptive SJF could be explored.
- ***Advanced Visualizations****:* More advanced visualizations such as color-coded Gantt charts could be incorporated for a better user experience.

GITHUB LINK: Click Here