****

**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)

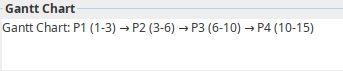
***3.2. Algorithm Implementation***

* 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.

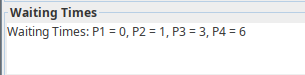
***3.3. Output Results***

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

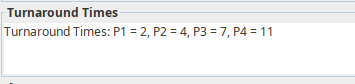
* ***Gantt Chart***: A visual timeline showing the execution order of processes.



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



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



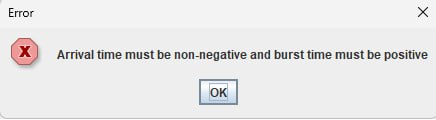
* ***Average Waiting Time***: The mean waiting time for all processes.
* ***Average Turnaround Time***: The mean turnaround time for all processes.



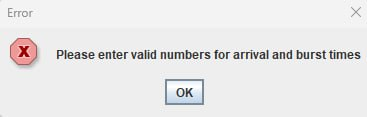
***3.4. Error Handling***

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

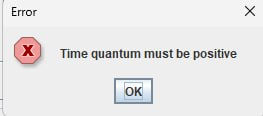
* ***Negative burst times***: An error message is displayed if a burst time is negative.



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



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



***3.5. Time Complexity***

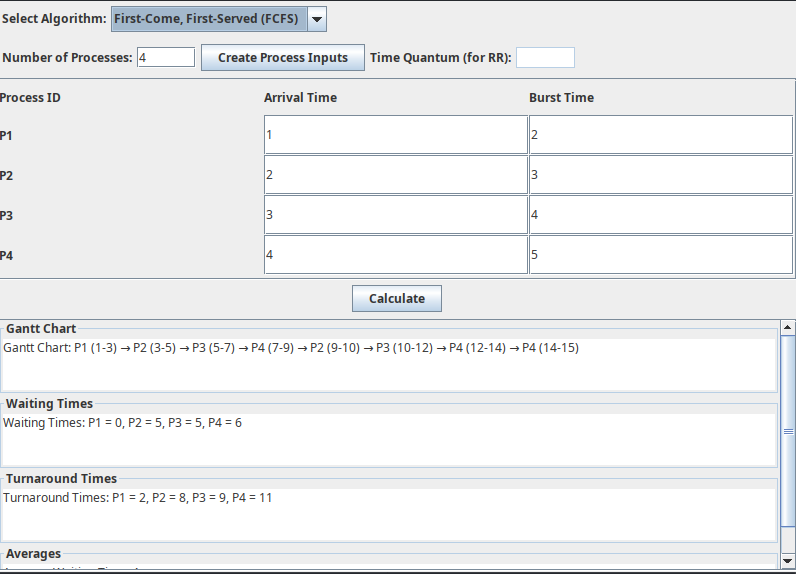
The time complexity of each scheduling algorithm is as follows:

* ***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.

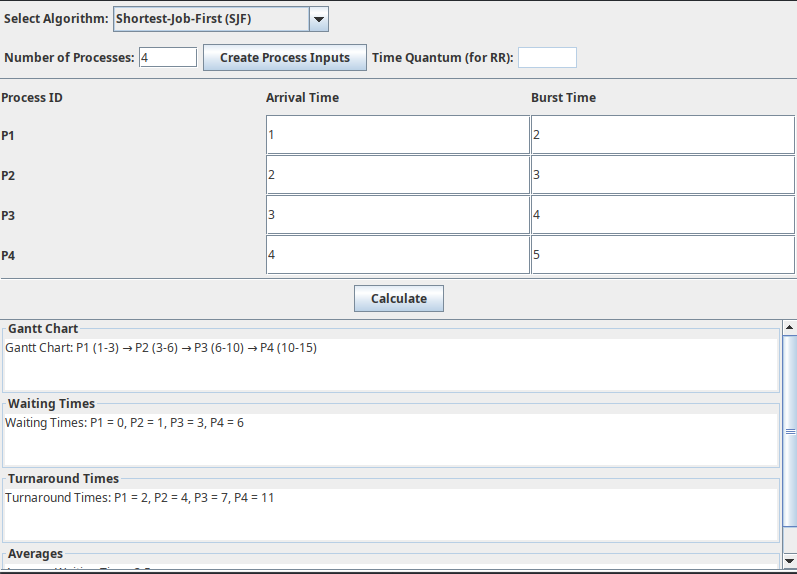
***4. Results and Evaluation***

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.

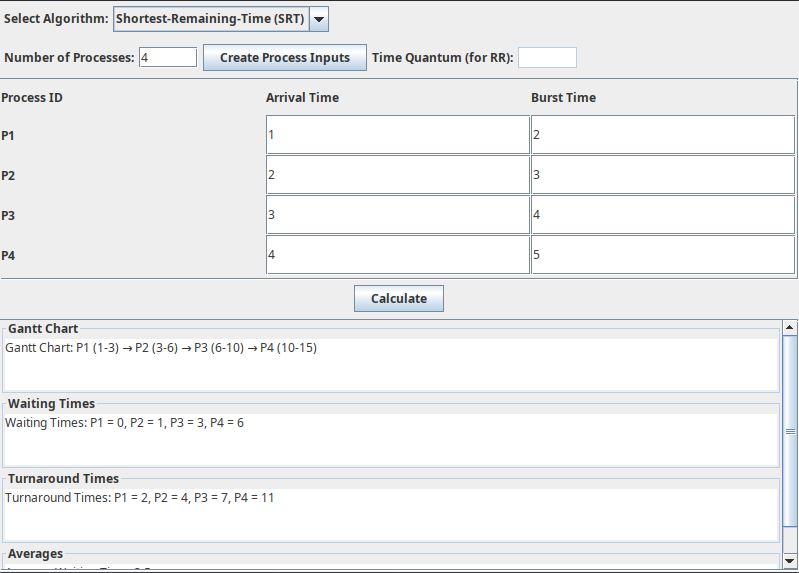
***Sample Output for FCFS***:



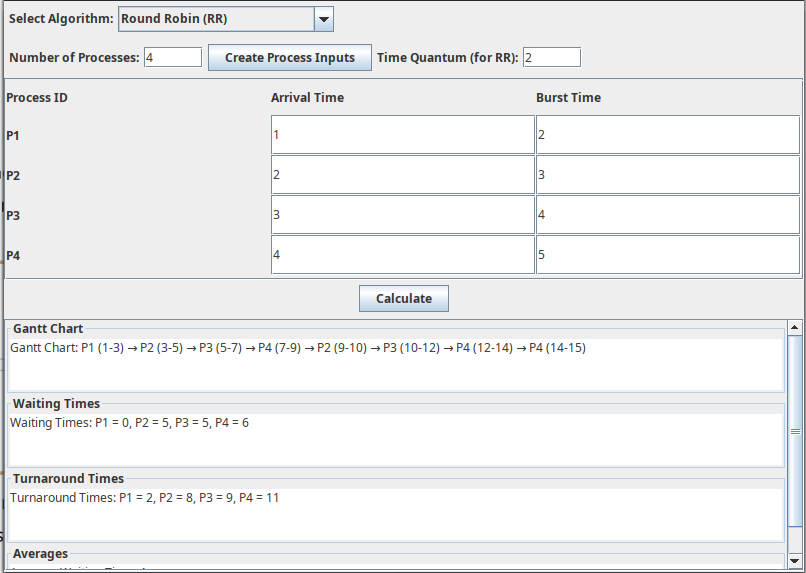
***Sample Output for SJF:***



***Sample Output for SRT***:



***Sample Output for RR*:**



***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](https://github.com/Bunheang360/IDT100048_Chheng_Bunheang_CPU_Scheduling.git)