CPU SCHEDULING

OS Project

AHMED BADAWY 2200091
MOHAMED ABOLYAZEED 2200337
MAHMOUD HAZEM 2200098
ABDELRAOUF DAHY 2102317

INTRODUCTION

This program simulates a basic CPU scheduling system, where users can manage and schedule processes based on various algorithms such as **FCFS (First Come, First Served)**, **SJF (Shortest Job First)**, **Priority Scheduling**, and **Round Robin**.

Key Features:

1- Process Management:

Users can input processes by specifying their CPU time, and the program automatically adds them to a table. Each process is given an ID (e.g., p1, p2) and initialized with waiting time and turnaround time set to 0.

2- Dynamic Table:

The program maintains a table of processes that gets updated as processes are added. For each process, the table tracks:

- Process ID
- CPU Time
- Waiting Time
- Turnaround Time
- (Optional) Priority (for priority-based algorithms)

3- Scheduling Algorithms:

Once processes are added, users can apply different scheduling algorithms to manage how the CPU processes tasks. Each algorithm calculates the waiting time and turnaround time for each process and displays an average waiting time.

4- Results Display:

After scheduling the processes using an algorithm, the program calculates and displays the average waiting time and average turnaround time, offering insights into the efficiency of the scheduling approach.

This tool helps in visualizing how different CPU scheduling algorithms manage tasks and optimize system performance based on various factors.

Types of CPU Scheduling Algorithms

1- First-Come, First-Served (FCFS):

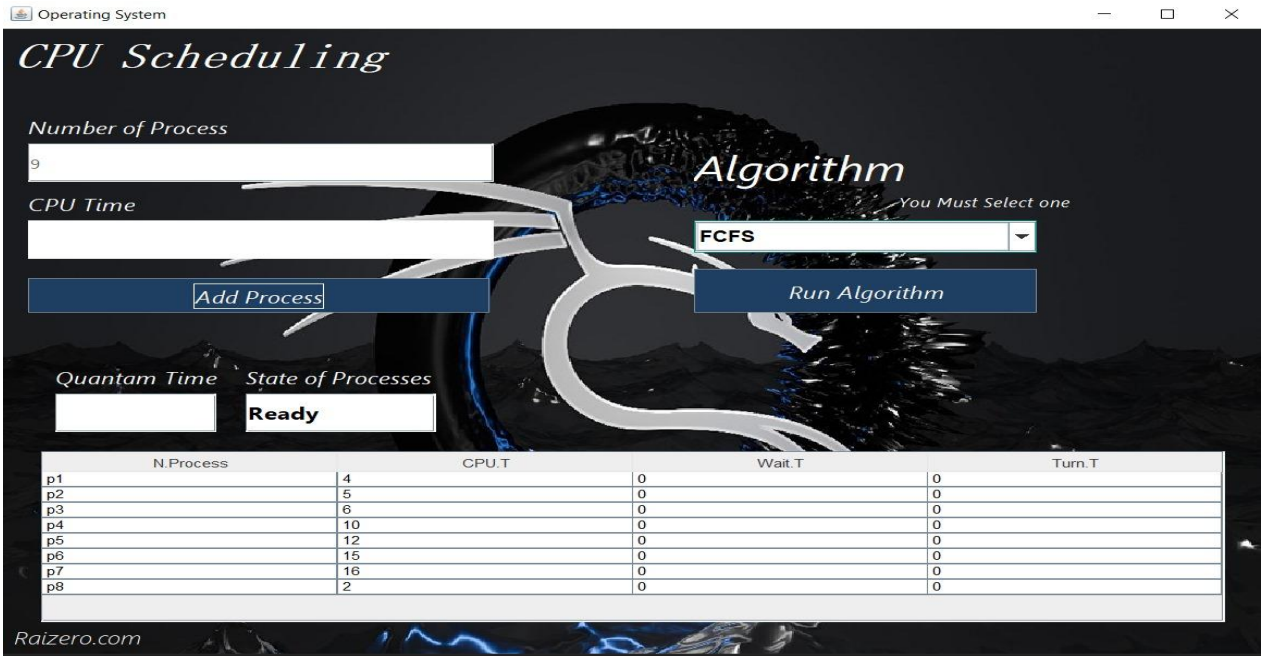
This code implements the First-Come, First-Served (FCFS) scheduling algorithm. It processes a list of tasks by calculating the CPU time, waiting time, and turnaround time for each task:

CPU Time: The algorithm starts by extracting the CPU time for each task from a data model.

Waiting Time: The waiting time for each task is calculated based on the cumulative sum of the CPU times of the previous tasks. The first task has a waiting time of 0.

Turnaround Time: The turnaround time for each task is the sum of its waiting time and CPU time.

Results: The calculated waiting times and turnaround times are displayed, and the average waiting time is shown.


Operating System
CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
FCFS
Run Algorithm
Quantam Time
State of Processes
Ready
N.Process CPU.T Wait.T Turn.T
p1 4 0 0
p2 5 0 0
p3 6 0 0
p4 10 0 0
p5 12 0 0
p6 15 0 0
p7 16 0 0
p8 2 0 0
Raizero.com


CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
FCFS
Result
in FCFS]
Average Waiting Time :26.25
average Trun Around Time :35.0
OK
Quantam Time
State of Processes
Running
N.Process CPU.T Wait.T Turn.T
p1 4 0 4
p2 5 4 9
p3 6 9 15
p4 10 15 25
p5 12 25 37
p6 15 37 52
p7 16 52 68
p8 2 68 70

2- Shortest Job First (SJF):

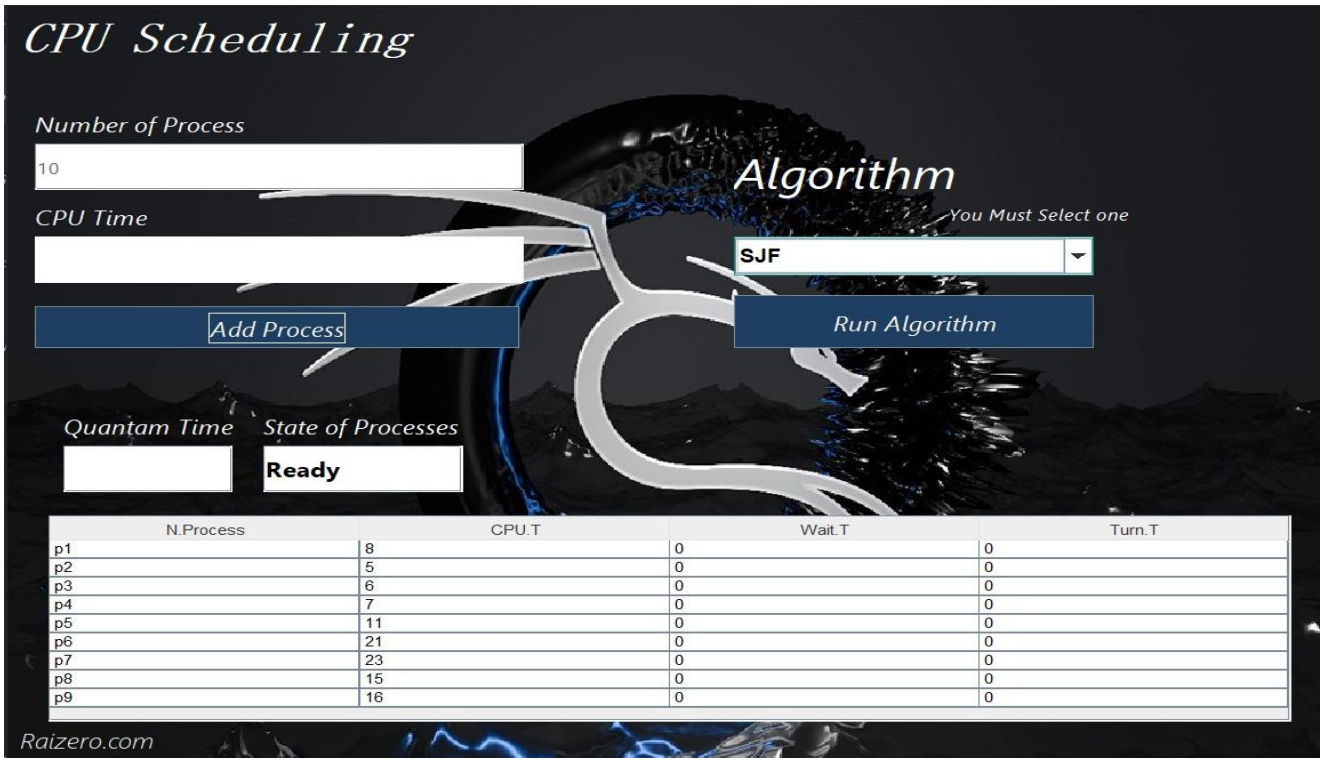
Task Setup: Each tasks CPU time, waiting time, and turnaround time are initialized and stored in a list.

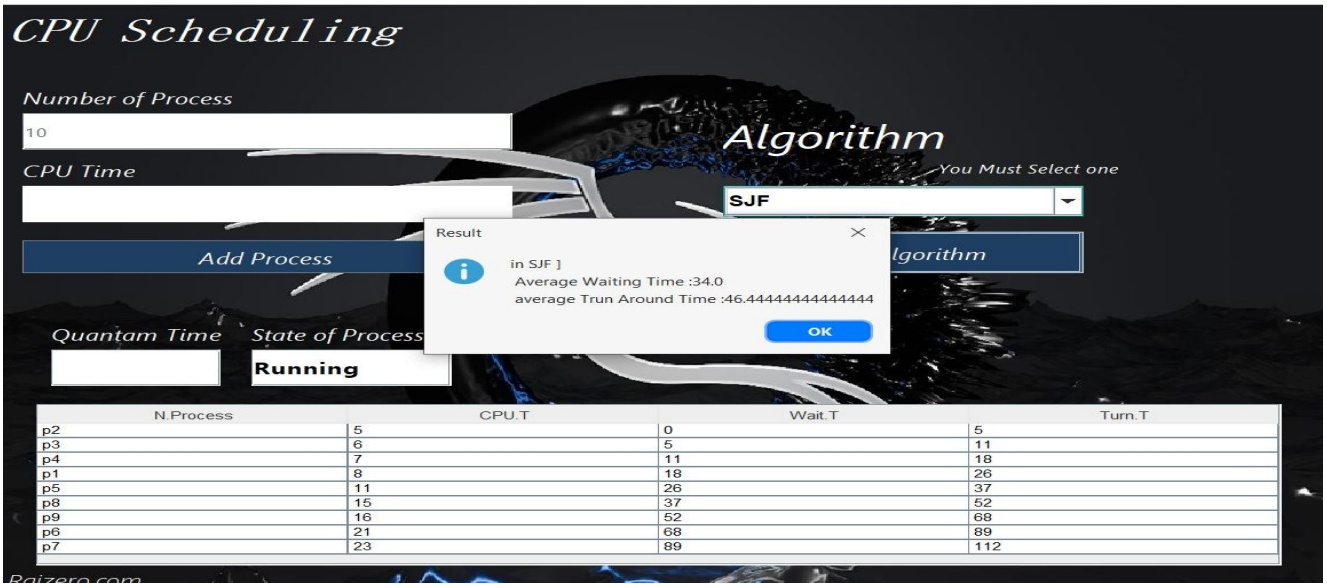
Sorting by CPU Time: The tasks are sorted by CPU time (shortest first) to ensure quicker tasks are processed first.

Time Calculation: The waiting time and turnaround time are calculated as each task is processed, with the current time being updated progressively.

Result Display: The table with updated times is shown, and the average waiting time for all tasks is calculated and displayed.

This approach ensures that the shortest tasks are prioritized, reducing waiting time for all processes.





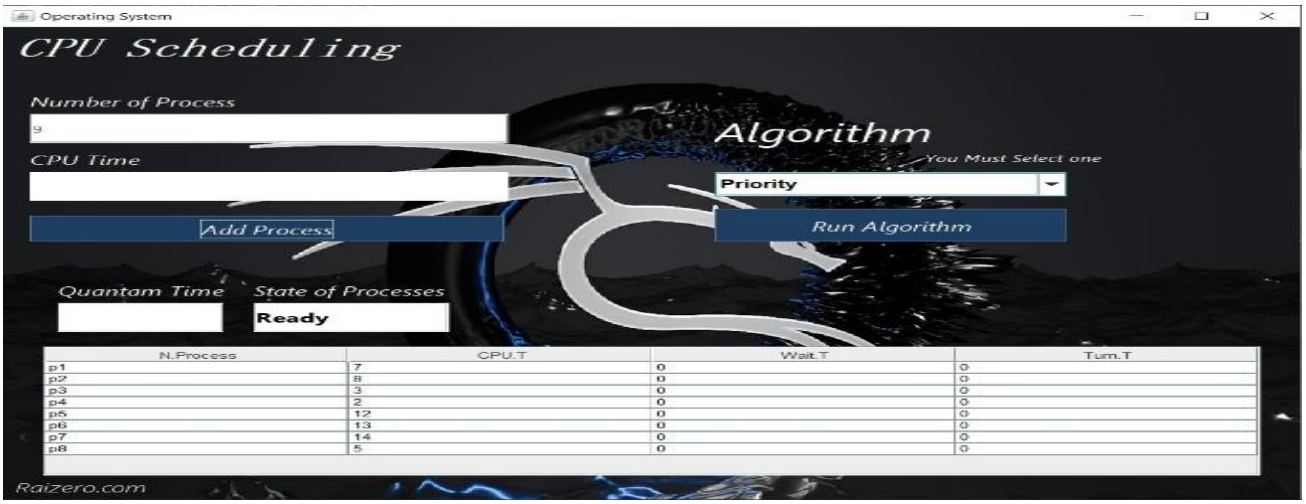
3- Priority Scheduling:

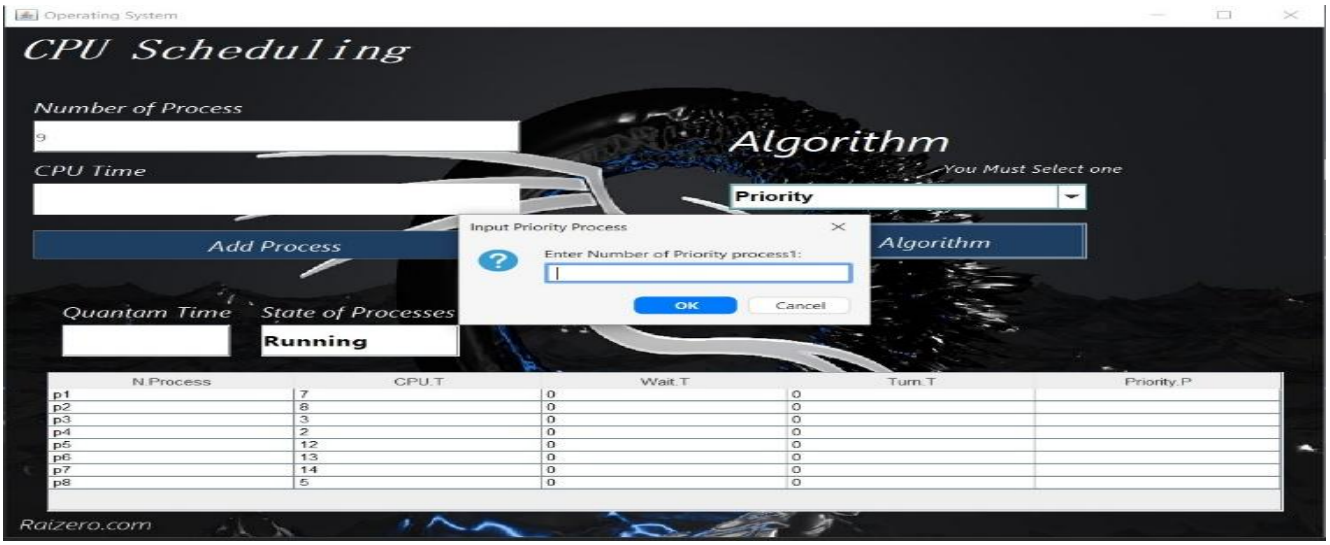
It prompts the user to input a priority value for each process and adds a new "Priority" column to the table.

The processes are then sorted by priority, with the lowest priority number being processed first.

Waiting time and turnaround time are calculated for each process based on the sorted order, with the current time being updated as each task is processed.

Finally, the updated task data is displayed in the table, and the average waiting time for all processes is calculated and shown.


Operating System
CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
Priority
Run Algorithm
Quantam Time
State of Processes
Ready
N.Process CPU.T Wait.T Turn.T
p1 7 0 0
p2 8 0 0
p3 3 0 0
p4 2 0 0
p5 12 0 0
p6 13 0 0
p7 14 0 0
p8 5 0 0
Raizero.com


Operating System
CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
Priority
Input Priority Process
Enter Number of Priority process1:
OK
Cancel
Algorithm
Quantam Time
State of Processes
Running
N.Process CPU.T Wait.T Turn.T Priority.P
p1 7 0 0
p2 8 0 0
p3 3 0 0
p4 2 0 0
p5 12 0 0
p6 13 0 0
p7 14 0 0
p8 5 0 0
Raizero.com


CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
Priority
Result
in Priority Algorithm]
Average Waiting Time :28.25
average Trun Around Time :36.25
OK
Quantam Time
State of Processes
Running
N.Process CPU.T Wait.T Turn.T Priority.P
p4 2 0 2 1
p5 12 2 14 2
p2 8 14 22 3
p1 7 22 29 4
p6 13 29 42 5
p7 14 42 56 6
p8 5 56 61 7
p3 3 61 64 8
Raizero.com

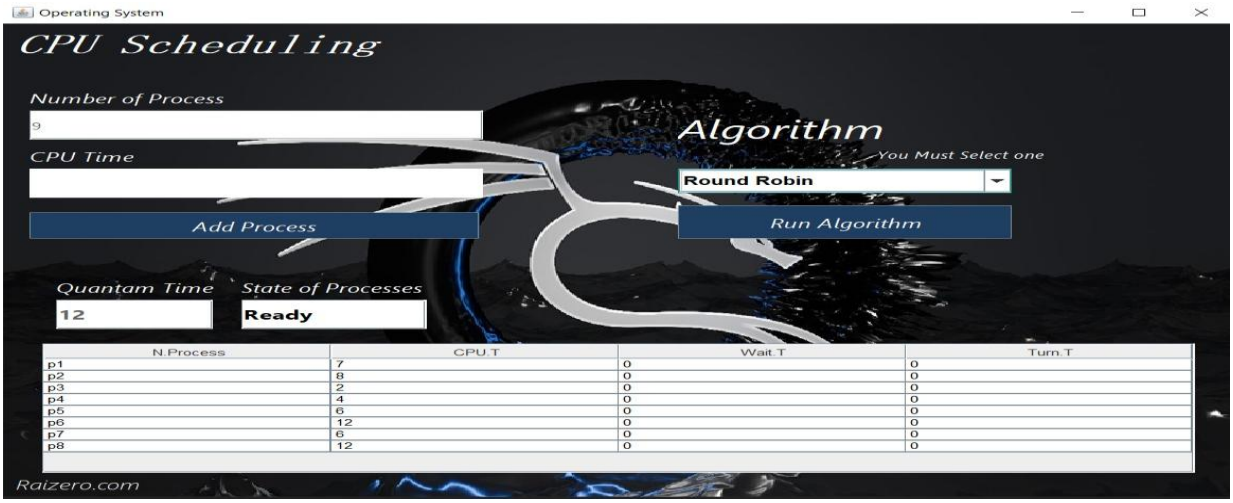
Round Robin:

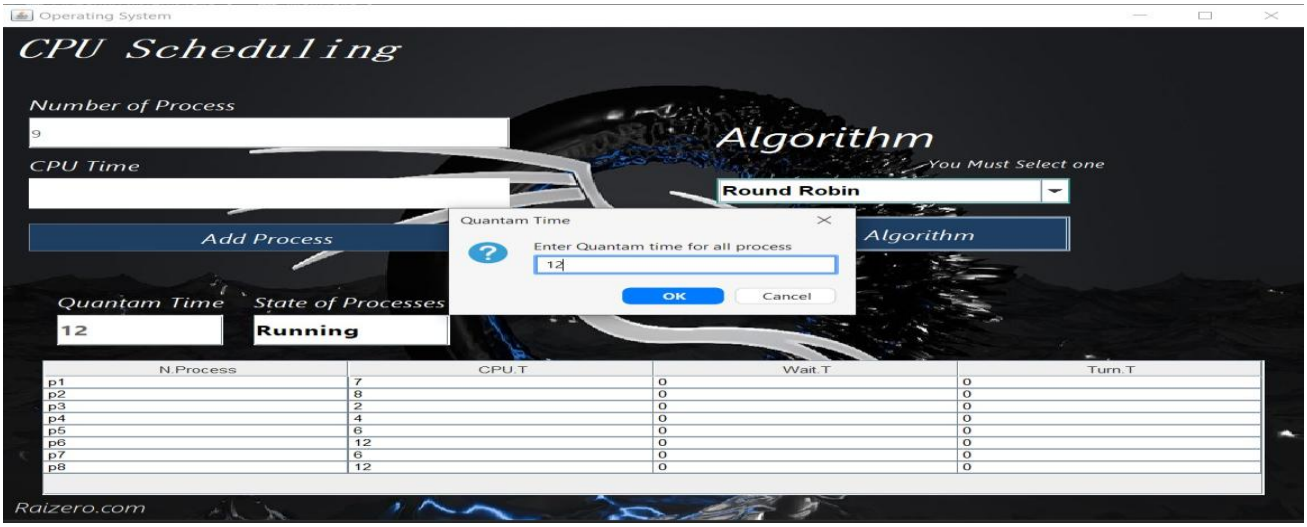
It prompts the user to input a quantum time and then initializes the processes with their CPU time, waiting time, and turnaround time.

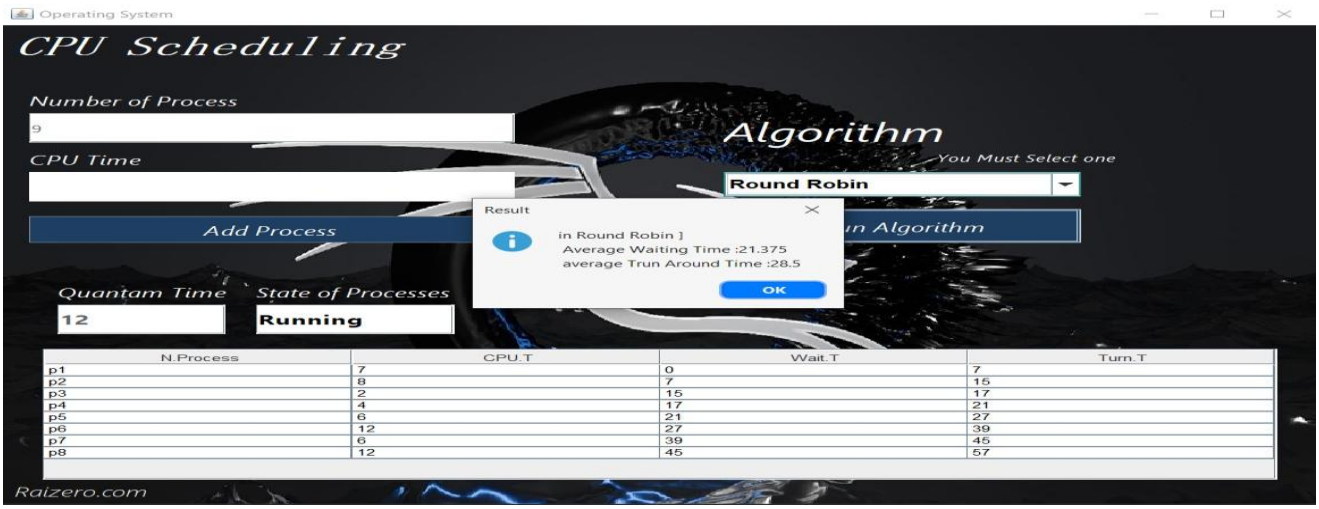
The processes are executed in a cyclic order, with each process getting a turn to run for the specified quantum time.

If a process requires more time than the quantum, it continues after other processes, otherwise, it's marked as finished, and its waiting time and turnaround time are calculated.

The process data is updated and displayed, and the average waiting time is calculated and shown after all processes are completed.






Operating System
CPU Scheduling
Number of Process
9
CPU Time
Add Process
Algorithm
You Must Select one
Round Robin
Result
in Round Robin]
Average Waiting Time :21.375
average Trun Around Time :28.5
OK
Quantam Time
12
State of Processes
Running
N.Process
CPU.T
Wait.T
Turn.T
p1 7 0 7
p2 8 7 15
p3 2 15 17
p4 4 17 21
p5 6 21 27
p6 12 27 39
p7 6 39 45
p8 12 45 57
Raizero.com

Thanks