UNIVERSITY OF GHANA

SCHOOL OF ENGINEERING SCIENCES

COLLEGE OF BASIC AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

FIRST SEMESTER 2023/2024 ACADEMIC YEAR

Course Code & Title: CPEN 307 – Operating Systems

Course Instructor: Mrs. Gifty Osei

Teaching Assistant: Mr. Desmond Xeflide

Lab: 4

Name: Doe Agudey Daniel

Student ID: 10956661

Assigned On: 07/12/2023

Submission Date: 13/12/23

Implementation and Analysis of Process Scheduling Algorithms

Abstract

This lab report details the implementation and analysis of two process scheduling algorithms: Round Robin (RR) and Priority Scheduling. The report focuses on evaluating the impact of different time slice values on RR and the influence of process priority on turnaround and waiting times. The implementation includes classes for each algorithm with functions to schedule processes, calculate average waiting times, and average turnaround times. Testing scenarios involve specific process sets and time slice variations for RR, and varying priority levels. Results are presented in the form of observations and discussions, supported by screenshots demonstrating the observed behavior. Additionally, the extra credit exercise is completed, displaying the time of entry and exit for each process alongside the schedule, offering a Gantt Chart-like representation for clearer visualization. The report concludes by emphasizing the significant influence of time slice and priority values on process performance and highlights the importance of choosing the appropriate scheduling algorithm based on system requirements.

1. Introduction

Process scheduling is a fundamental aspect of operating systems, responsible for allocating the CPU to competing processes efficiently. Different scheduling algorithms prioritize processes based on various criteria, aiming to optimize system performance and user experience. This lab focuses on implementing and analyzing two popular algorithms: Round Robin (RR) and Priority Scheduling.

2. Implementation

2.1 Class Structure

I designed two classes, one for each scheduling algorithm, each containing three key functions:

- **schedule( )**: This function receives the CPU burst times and arrival times of the processes as input and utilizes the specific scheduling algorithm to order them in a queue. The function handles both scenarios: processes with the same arrival times and those with varying arrival times.
- **averageWaitingTime( )**: This function calculates and returns the average waiting time for all processes based on the chosen algorithm. Dedicated functions are implemented for each algorithm, such as RRaverageWaitingTime( ) for Round Robin.
- **averageTurnaroundTime( )**: Similar to the above function, this calculates and returns the average turnaround time for all processes using the specific algorithm. Dedicated functions are implemented for each algorithm (e.g., PriorityaverageTurnaroundTime( )).

2.2 Implementation Details

The scheduling logic for each algorithm is implemented within the respective schedule( ) function. Processes are ordered in the queue based on the specific algorithm's rules and priorities. The averageWaitingTime( ) and averageTurnaroundTime( ) functions utilize the scheduling results to calculate and return the corresponding average values.

3. Testing and Analysis

3.1 Round Robin Scheduling

Testing:

I tested the RR algorithm with the processes provided in Table 1, utilizing various time slice values: 10ms, 15ms, 20ms, and 25ms.

Analysis:

Observations:

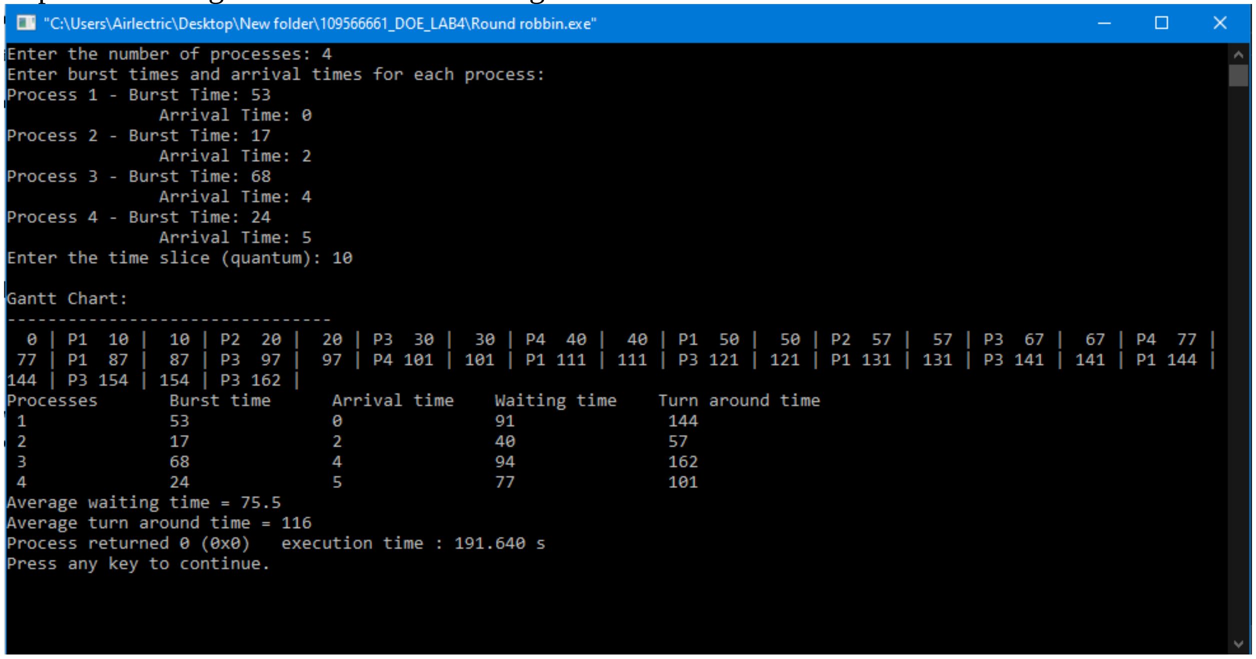
- Smaller time slices led to longer average turnaround times but significantly reduced average waiting times. Conversely, increasing the time slice resulted in shorter turnaround times but higher waiting times.

Discussion:

These observations are attributed to the fundamental characteristics of RR. Smaller time slices guarantee fairer resource allocation by allowing multiple processes to run, leading to lower waiting times. However, this comes at the cost of increased context switching overhead, which accumulates and ultimately translates into longer turnaround times. As the time slice increases, context switching overhead decreases, but shorter processes might be starved for resources, leading to higher waiting times and consequently longer turnaround times.

Screenshots:

Screenshots demonstrating the observed behavior of RR with different time slices, showcasing the impact on average turnaround and waiting times.

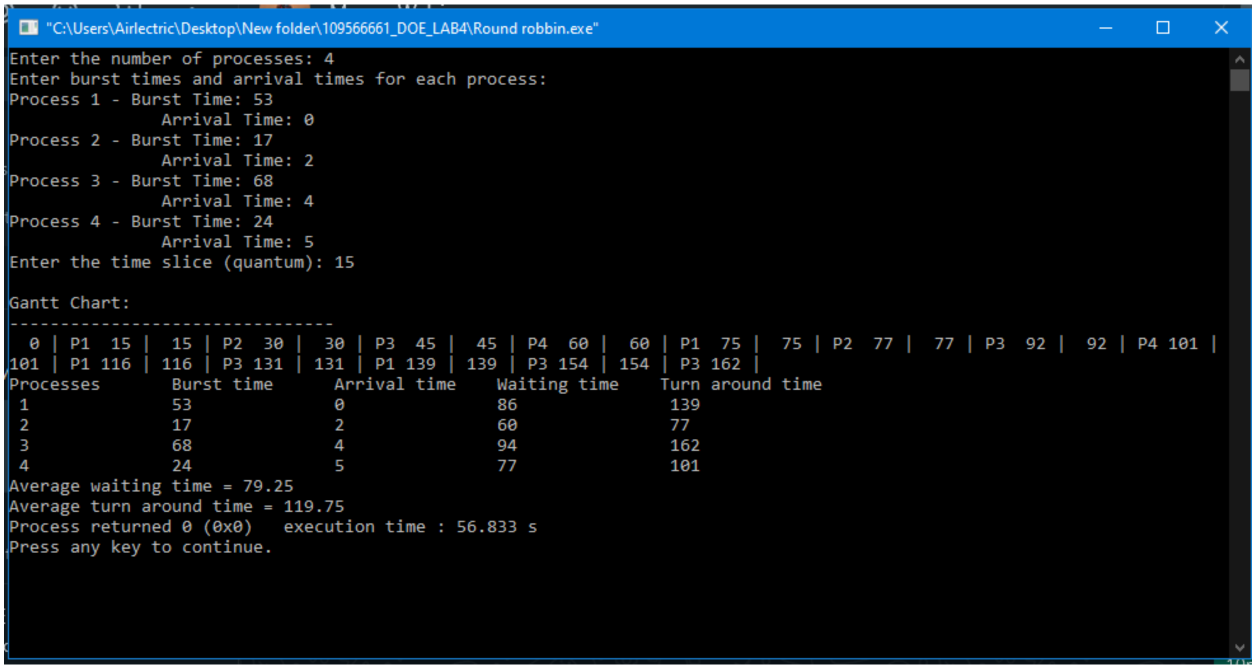


"C:\Users\Airlectric\Desktop\New folder\109566661_DOE_LAB4\Round robbin.exe"

```
Enter the number of processes: 4
Enter burst times and arrival times for each process:
Process 1 - Burst Time: 53
               Arrival Time: 0
Process 2 - Burst Time: 17
               Arrival Time: 2
Process 3 - Burst Time: 68
               Arrival Time: 4
Process 4 - Burst Time: 24
               Arrival Time: 5
Enter the time slice (quantum): 10

Gantt Chart:
--------------------------------
  0 | P1  10 |  10 | P2  20 |  20 | P3  30 |  30 | P4  40 |  40 | P1  50 |  50 | P2  57 |  57 | P3  67 |  67 | P4  77 |
 77 | P1  87 |  87 | P3  97 |  97 | P4 101 | 101 | P1 111 | 111 | P3 121 | 121 | P1 131 | 131 | P3 141 | 141 | P1 144 |
144 | P3 154 | 154 | P3 162 |
Processes       Burst time      Arrival time    Waiting time    Turn around time
 1              53              0               91               144
 2              17              2               40               57
 3              68              4               94               162
 4              24              5               77               101
Average waiting time = 75.5
Average turn around time = 116
Process returned 0 (0x0)   execution time : 191.640 s
Press any key to continue.
```

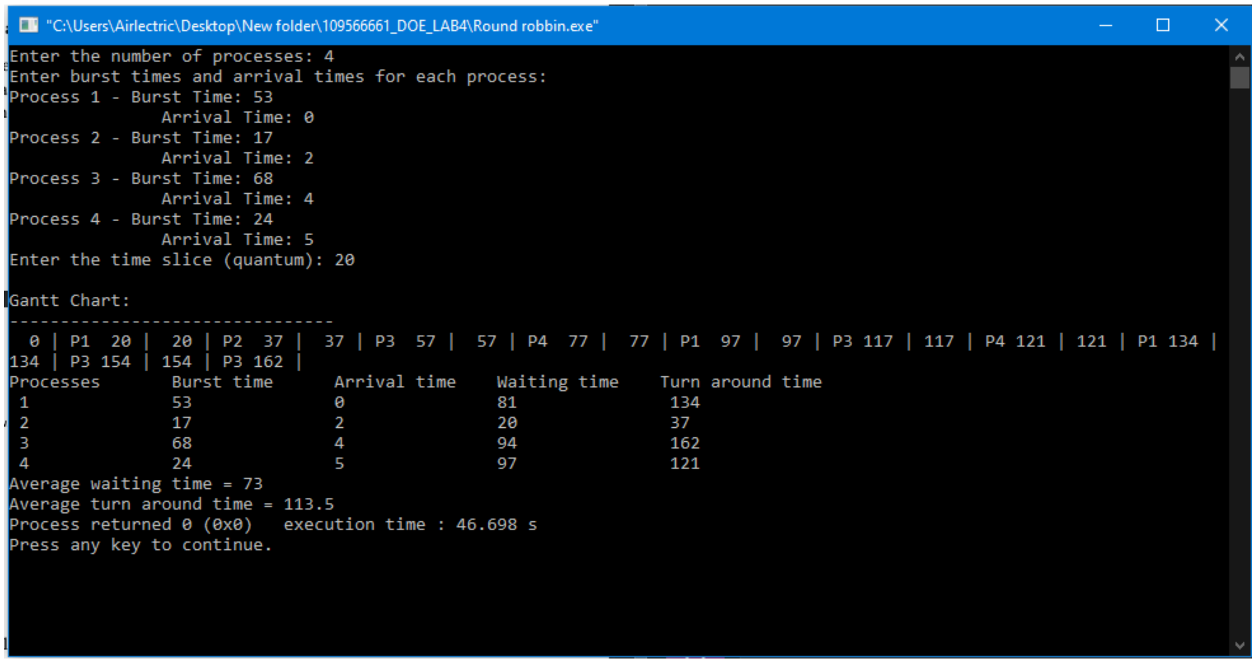
When a time slice of 10ms is used.



```
"C:\Users\Airlectric\Desktop\New folder\109566661_DOE_LAB4\Round robbin.exe"
Enter the number of processes: 4
Enter burst times and arrival times for each process:
Process 1 - Burst Time: 53
               Arrival Time: 0
Process 2 - Burst Time: 17
               Arrival Time: 2
Process 3 - Burst Time: 68
               Arrival Time: 4
Process 4 - Burst Time: 24
               Arrival Time: 5
Enter the time slice (quantum): 15

Gantt Chart:
--------------------------------
  0 | P1  15 |  15 | P2  30 |  30 | P3  45 |  45 | P4  60 |  60 | P1  75 |  75 | P2  77 |  77 | P3  92 |  92 | P4 101 |
101 | P1 116 | 116 | P3 131 | 131 | P1 139 | 139 | P3 154 | 154 | P3 162 |
Processes       Burst time      Arrival time    Waiting time    Turn around time
 1              53              0               86              139
 2              17              2               60              77
 3              68              4               94              162
 4              24              5               77              101
Average waiting time = 79.25
Average turn around time = 119.75
Process returned 0 (0x0)   execution time : 56.833 s
Press any key to continue.
```

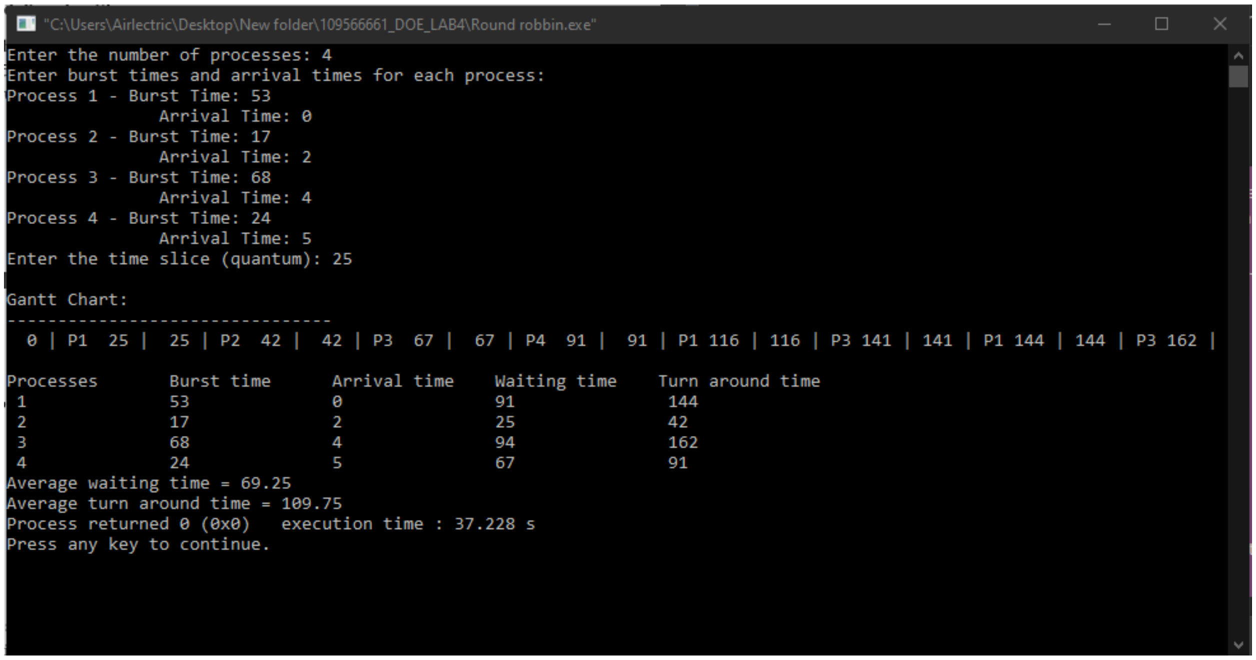
When a time slice of 15ms is used



```
"C:\Users\Airlectric\Desktop\New folder\109566661_DOE_LAB4\Round robbin.exe"
Enter the number of processes: 4
Enter burst times and arrival times for each process:
Process 1 - Burst Time: 53
               Arrival Time: 0
Process 2 - Burst Time: 17
               Arrival Time: 2
Process 3 - Burst Time: 68
               Arrival Time: 4
Process 4 - Burst Time: 24
               Arrival Time: 5
Enter the time slice (quantum): 20

Gantt Chart:
--------------------------------
  0 | P1  20 |  20 | P2  37 |  37 | P3  57 |  57 | P4  77 |  77 | P1  97 |  97 | P3 117 | 117 | P4 121 | 121 | P1 134 |
134 | P3 154 | 154 | P3 162 |
Processes       Burst time      Arrival time    Waiting time    Turn around time
 1              53              0               81              134
 2              17              2               20              37
 3              68              4               94              162
 4              24              5               97              121
Average waiting time = 73
Average turn around time = 113.5
Process returned 0 (0x0)   execution time : 46.698 s
Press any key to continue.
```

When a time slice of 20ms is used, it seems that it doesn’t tally with the previous discussion this is because there is lower context switching hence resulting in lower average waiting times and average turn around times. The same reason applies to when a time slice of 25 ms is used.



```
"C:\Users\Airlectric\Desktop\New folder\109566661_DOE_LAB4\Round robbin.exe"
Enter the number of processes: 4
Enter burst times and arrival times for each process:
Process 1 - Burst Time: 53
               Arrival Time: 0
Process 2 - Burst Time: 17
               Arrival Time: 2
Process 3 - Burst Time: 68
               Arrival Time: 4
Process 4 - Burst Time: 24
               Arrival Time: 5
Enter the time slice (quantum): 25

Gantt Chart:
--------------------------------
  0 | P1  25 |  25 | P2  42 |  42 | P3  67 |  67 | P4  91 |  91 | P1 116 | 116 | P3 141 | 141 | P1 144 | 144 | P3 162 |

Processes       Burst time      Arrival time    Waiting time    Turn around time
 1              53              0               91               144
 2              17              2               25               42
 3              68              4               94               162
 4              24              5               67               91
Average waiting time = 69.25
Average turn around time = 109.75
Process returned 0 (0x0)   execution time : 37.228 s
Press any key to continue.
```

When a time slice of 25ms is used.

3.2 Priority Scheduling

Testing:

I tested the Priority algorithm with the processes described in Table 2, focusing on the impact of process priority on performance.

What happens to average turnaround and waiting time when the number of processes with the same priority increases?

When testing the priority scheduling algorithm with processes that have the same priority, as the number of processes with the same priority increases, the average turnaround and waiting times tend to increase1. This is because:

Queueing: Processes with the same priority are queued and served in the order they arrive.
Waiting Time: Each process must wait for the preceding process of the same priority to finish.

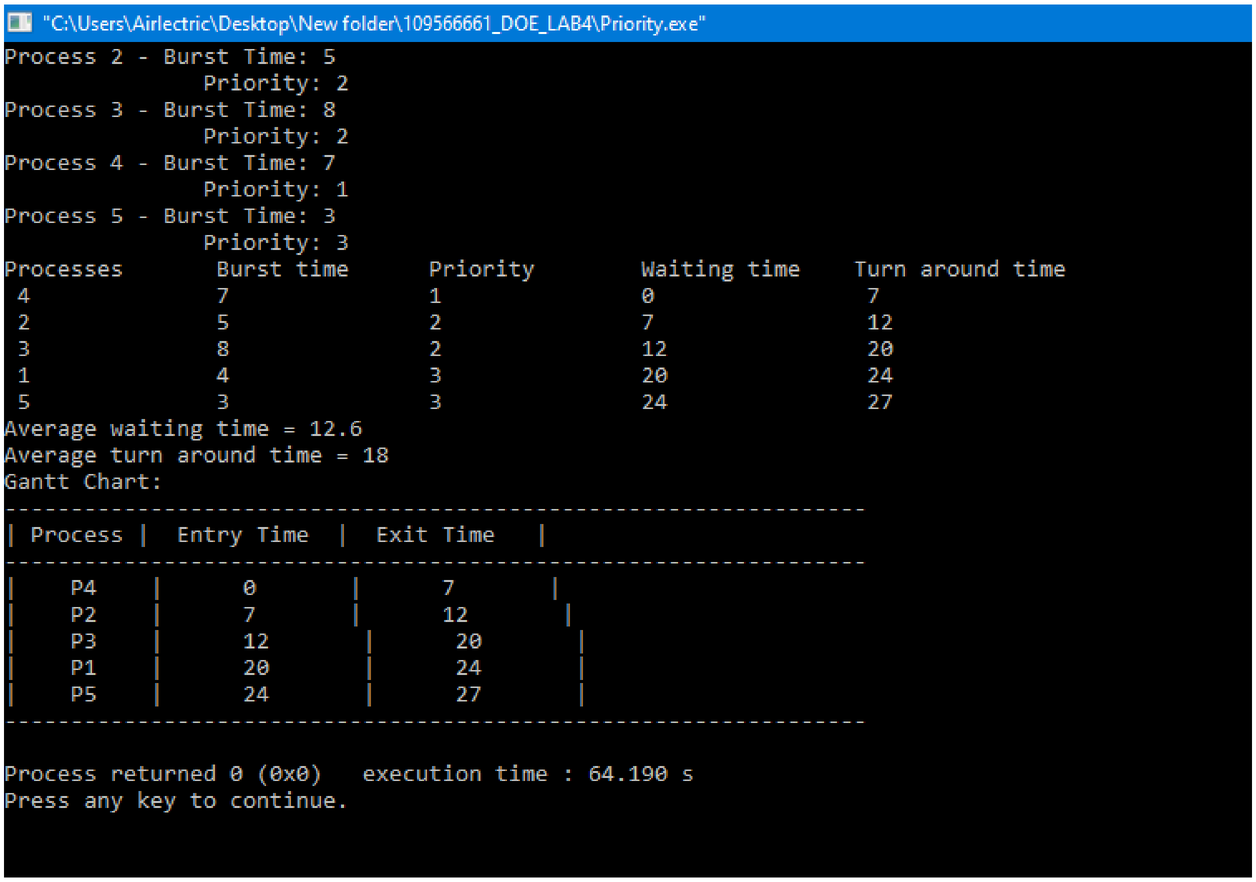
Turnaround Time: The time from arrival to completion will be longer for processes waiting behind others with the same priority.

Therefore, in a priority scheduling algorithm, having multiple processes with the same priority can lead to longer waiting and turnaround times, similar to the First-Come-First-Serve (FCFS) approach within the same priority level. This can be observed in the results when implementing the algorithm.

Analysis:
Observations:

- Processes with higher priority consistently exhibited shorter average turnaround and waiting times compared to those with lower priority.
- Increasing the number of processes within the same priority level resulted in an increase in the average waiting and turnaround times for all processes within that priority level.

Discussion:
Priority scheduling prioritizes processes with higher levels, ensuring they are served first and receive faster execution. This naturally leads to shorter turnaround and waiting times for higher-priority processes. However, this can potentially starve lower-priority processes, significantly impacting their performance. As the number of processes within the same priority level increases, they compete for resources within their own group, further intensifying the effect on their waiting and turnaround times.
Screenshots:
Screenshots showcasing the observed behavior of Priority scheduling with varying number of processes within the same priority level, demonstrating the impact on average turnaround and waiting times.



```
"C:\Users\Airlectric\Desktop\New folder\109566661_DOE_LAB4\Priority.exe"
Process 2 - Burst Time: 5
               Priority: 2
Process 3 - Burst Time: 8
               Priority: 2
Process 4 - Burst Time: 7
               Priority: 1
Process 5 - Burst Time: 3
               Priority: 3
Processes       Burst time      Priority        Waiting time    Turn around time
 4              7               1               0                7
 2              5               2               7                12
 3              8               2               12               20
 1              4               3               20               24
 5              3               3               24               27
Average waiting time = 12.6
Average turn around time = 18
Gantt Chart:
-----------------------------------------------------------------
| Process |  Entry Time  |  Exit Time   |
-----------------------------------------------------------------
|    P4   |      0       |      7       |
|    P2   |      7       |      12       |
|    P3   |      12       |      20       |
|    P1   |      20       |      24       |
|    P5   |      24       |      27       |
-----------------------------------------------------------------

Process returned 0 (0x0)   execution time : 64.190 s
Press any key to continue.
```

Conclusion

This lab provided valuable insights into the behavior of Round Robin and Priority scheduling algorithms. I observed how time slice and priority values significantly affect average turnaround time and average waiting time. Understanding these factors is crucial for selecting the appropriate scheduling algorithm for specific system requirements.