****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Semester: (Spring, Year:2024), B.Sc. in CSE (Day)**

**Lab Report NO #03**

**Course Title:** Operating System Lab

**Course Code:** CSE 310  **Section:** 221 D6

**Lab Experiment Name:** CPU Schedule Algorithms to find Turnaround Time and Waiting Time.

**Student Details**

| **Name** | | **ID** |
| --- | --- | --- |
| **1.** | Tanvir Ahmed | 221002461 |

**Lab Date : 25/3/2024**

**Submission Date : 1/4/2024**

**Course Teacher’s Name : Jarin Tasnim Tonvi**

| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |
| --- |

**1. TITLE OF THE LAB REPORT EXPERIMENT**

CPU scheduling algorithm to find turnaround time and waiting time.

**2. OBJECTIVES/AIM**

* To gather knowledge of CPU scheduling algorithms.
* To implement a problem based on the CPU scheduling algorithm, the condition is priority>sjf>fcfs.

**3. PROCEDURE**

* At first, take the number of processes and then burst time and priority for the process from the user.
* After taking input, sort the priority. That means the process has the highest priority then we execute the process first.
* If several priorities are the same then sort based on the shortest burst time.
* If the burst time is also the same then the first one entered the array will execute first.
* After that, calculating waiting time and turnaround time and at last, showing all results.

**4. IMPLEMENTATION**

Source Code:

1. /\* SJF CPU Scheduling Algorithm code \*/
2. #include<stdio.h>
3. #include<conio.h>
4. **main**()
5. {
6. int p[100], bt[100], wt[100], tat[100], n, pr[100];
7. float wtavg, tatavg;
8. **printf**("\n Enter the number of processes: ");
9. **scanf**("%d", &n);
10. **printf**("\n");
11. int i;
12. **for**(i=0; i<n; i++)
13. {
14. p[i]=i+1;
15. **printf**(" Enter Burst Time for Process P%d: ", i+1);
16. **scanf**("%d", &bt[i]);
17. **printf**(" Enter Priority for Process P%d: ", i+1);
18. **scanf**("%d", &pr[i]);
19. }
20. int j, temp;
21. **for**(i=0; i<n; i++)
22. {
23. **for**(j=i+1; j<n; j++)
24. {
25. **if**(pr[i]>pr[j])
26. {
27. temp=pr[i];
28. pr[i]=pr[j];
29. pr[j]=temp;
30. temp=bt[i];
31. bt[i]=bt[j];
32. bt[j]=temp;
33. temp=p[i];
34. p[i]=p[j];
35. p[j]=temp;
36. }
37. **else** **if**(pr[i] == pr[j])
38. {
39. **if**(bt[i]>bt[j])
40. {
41. temp=bt[i];
42. bt[i]=bt[j];
43. bt[j]=temp;
44. temp=p[i];
45. p[i]=p[j];
46. p[j]=temp;
47. }
48. }
49. }
50. }
51. wt[0] = wtavg = 0;
52. tat[0] = tatavg = bt[0];
53. **for**(i=1; i<n; i++)
54. {
55. wt[i] = tat[i-1];
56. tat[i] = wt[i] + bt[i];
57. wtavg = wtavg + wt[i];
58. tatavg = tatavg + tat[i];
59. }
60. **printf**("\n\t PROCESS \tBURST TIME \tPRORITY \t WAITING TIME\t TURNAROUND TIME\n");
61. **for**(i=0; i<n; i++)
62. {
63. **printf**("\n\t P%d \t\t %d \t\t %d \t\t %d \t\t %d", p[i], bt[i], pr[i], wt[i], tat[i]);
64. }
65. **printf**("\n\n");
66. **printf**("Average Waiting Time: %f \n", wtavg/n);
67. **printf**("\n");
68. **printf**("Average Turnaround Time: %f \n", tatavg/n);
69. **printf**("\n");
70. }

**5. TEST RESULT / OUTPUT**

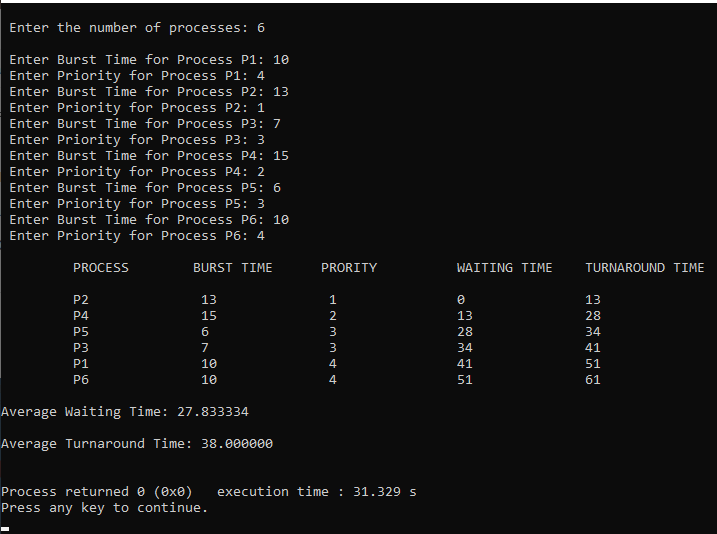
****

Fig1. Here is the sample input and output for the problem.

**6. ANALYSIS AND DISCUSSION**

* In this exercise, I implement a given problem using cpu scheduling algorithm.
* There is a condition, that is, the process will execute based on the priority. If the priority is the same then execute based on the shortest job first, if it is also the same then execute based on fcfs.
* I learned how to implement the problem based on fcfs and sjf in the previous class.
* I simply wrote the same code which I implemented in the previous lab class and then I just modified the code a little bit to match the exercise problem.
* If the priority is the same then I just run the logic of sjf.
* If the burst time is also the same then I stored the data in fcfs order. It will also execute in fcfs order.

**7. SUMMARY**

This lab exercise demonstrates the successful implementation of CPU scheduling problems based on the knowledge that I learned from the previous lab class. I knew how to implement sjf and fcfs, and I implemented the priority-based CPU scheduling algorithm on lab tasks. I merged all of them in this exercise and solved the problem. I solved the problem without any difficulties, yes I faced some syntax errors and I fixed those problems without any problems.