Ex. No.: 6b)
Date:

SHORTEST JOB FIRST

Aim:

To implement the Shortest Job First (SJF) scheduling technique

Algorithm:

1. Declare the structure and its elements.
2. Get number of processes as input from the user.
3. Read the process name, arrival time and burst time
4. Initialize waiting time, turnaround time & flag of read processes to zero. 5. Sort based on burst time of all processes in ascending order 6. Calculate the waiting time and turnaround time for each process. 7. Calculate the average waiting time and average turnaround time. 8. Display the results.

Program Code:

```
#include <stdio.h>

int main() {
   int n;
   printf("Enter no. of processes: ");
   scanf("%d", &n);

   int burst[2][n]; // burst[0] for Process ID, burst[1] for Burst Time
   int arr[n]; // Not used, so removed in corrected code

   for (int i = 0; i < n; i++) {
      printf("Enter burst time of process %d: ", i + 1);
      burst[0][i] = i + 1; // Storing process number
      scanf("%d", &burst[1][i]);
   }

   // Sorting based on Burst Time using Bubble Sort (SJF Non-Preemptive)
   for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - 1 - i; j++) {
         if (burst[1][j] > burst[1][j + 1]) { // Sort by burst time
            // Swap burst time
            int temp = burst[1][j];
            burst[1][j] = burst[1][j + 1];
            burst[1][j + 1] = temp;

            // Swap process ID accordingly
            temp = burst[0][j];
            burst[0][j] = burst[0][j + 1];
            burst[0][j + 1] = temp;
         }
      }
   }
```

```
    // Calculating Turnaround Time and Waiting Time
    int turn_around[n], waiting[n], avg_turn = 0, avg_wait = 0;

    turn_around[0] = burst[1][0]; // First turnaround time is just its burst time
    waiting[0] = 0; // First process has no waiting time

    for (int i = 1; i < n; i++) {
        turn_around[i] = burst[1][i] + turn_around[i - 1]; // TAT[i] = BT[i] + TAT[i-1]
        waiting[i] = turn_around[i] - burst[1][i]; // WT[i] = TAT[i] - BT[i]
    }

    // Calculating Average TAT & WT
    for (int i = 0; i < n; i++) {
        avg_turn += turn_around[i];
        avg_wait += waiting[i];
    }

    // Displaying the results
    printf("\nProcess\t Burst\t TurnAround\t Waiting\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t %d\t %d\t\t %d\n", burst[0][i], burst[1][i], turn_around[i], waiting[i]);
    }
}
```

Sample Output:
Enter the number of process:
4
Enter the burst time of the processes:
8 4 9 5

Process	Burst Time	Waiting Time	Turn Around Time
2	4	0	4
4	5	4	9
1	8	9	17
3	9	17	26

Average waiting time is: 7.5
Average Turn Around Time is:14.0

Output:

```
[student@localhost ~]$ ./a.out
Enter no. of processes: 4
Enter burst time of process 1: 8
Enter burst time of process 2: 4
Enter burst time of process 3: 9
Enter burst time of process 4: 5

Process  Burst   TurnAround      Waiting
2        4       4               0
4        5       9               4
1        8       17              9
3        9       26              17
Average waiting time is: 7.50
Average turn around time is: 14.00
[student@localhost ~]$
```

Result:
Program is executed successfully and output is verified.