Name:K.SaiKrishna

Reg-No: 192311106

4. Construct a scheduling program with C that selects the waiting process with the smallest execution time to execute next

Aim

To implement the Shortest Job Next (SJN) scheduling algorithm in C, which minimizes average waiting and turnaround times by selecting processes based on their burst times.

Algorithm

1. Input the number of processes and their burst times.
2. Sort the processes based on burst times in ascending order.
3. Execute processes sequentially based on the sorted order.
4. Calculate and display the average waiting time (AWT) and average turnaround time (ATAT).

Procedure

1. Input process details: process IDs and burst times.
2. Sort the processes by their burst times.
3. Calculate waiting time (WT) and turnaround time (TAT) for each process:
    - **WT** = (Cumulative sum of previous burst times).
    - **TAT** = WT + Burst Time.
4. Compute and display AWT and ATAT.

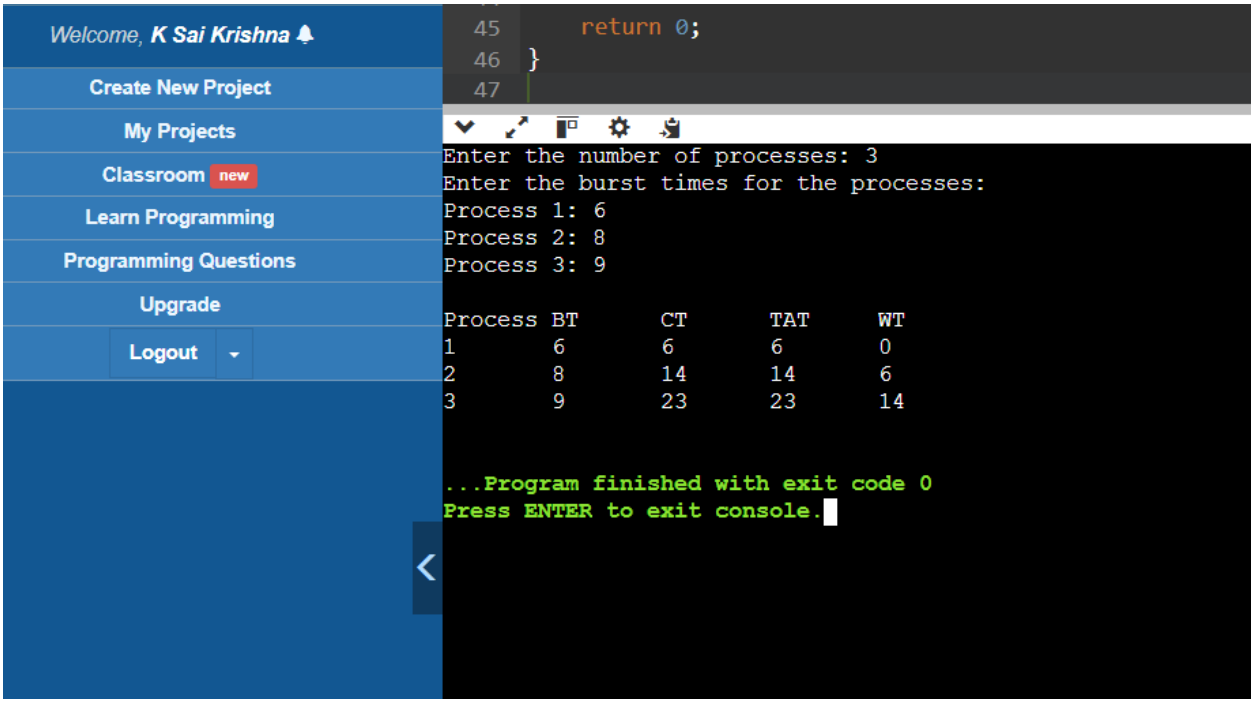
Code:

```
#include <stdio.h>

void main() {

   int n, i, j, temp;

   printf("Enter the number of processes: ");

   scanf("%d", &n);

int process[n], burst_time[n], waiting_time[n], turnaround_time[n];

   for (i = 0; i < n; i++) {

      printf("Enter burst time for process %d: ", i + 1);

      scanf("%d", &burst_time[i]);
```

```
        process[i] = i + 1;
    }
for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - i - 1; j++) {
            if (burst_time[j] > burst_time[j + 1]) {
                temp = burst_time[j];
                burst_time[j] = burst_time[j + 1];
                burst_time[j + 1] = temp;
                temp = process[j];
                process[j] = process[j + 1];
                process[j + 1] = temp;
            }
        }
    }
waiting_time[0] = 0;
    for (i = 1; i < n; i++) {
        waiting_time[i] = waiting_time[i - 1] + burst_time[i - 1];
    }
 float total_waiting_time = 0, total_turnaround_time = 0;
    for (i = 0; i < n; i++) {
        turnaround_time[i] = waiting_time[i] + burst_time[i];
        total_waiting_time += waiting_time[i];
        total_turnaround_time += turnaround_time[i];
```

```
    }  printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");

    for (i = 0; i < n; i++) {

        printf("%d\t%d\t\t%d\t\t%d\n", process[i], burst_time[i], waiting_time[i],
turnaround_time[i]);

    }

 printf("\nAverage Waiting Time: %.2f\n", total_waiting_time / n);

    printf("Average Turnaround Time: %.2f\n", total_turnaround_time / n);

}
```

Output:



Result

The program calculates and displays the **waiting time** and **turnaround time** for each process.

It computes and displays the **average waiting time** and **average turnaround time**.