**Name: Sheikh Ahaduzzaman**

**ID: 1711176112**

## FCFS Code :

```cpp
#include <iostream>
using namespace std;
void findWaitingTime(int processes[], int n, int bt[], int wt[])
{
    wt[0] = 0;
    for (int i = 1; i < n; i++)
        wt[i] = bt[i - 1] + wt[i - 1];
}
void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[])
{
    for (int i = 0; i < n; i++)
        tat[i] = bt[i] + wt[i];
}
void findavgTime(int processes[], int n, int bt[])
{
    int wt[n], tat[n], total_wt = 0, total_tat = 0;
    findWaitingTime(processes, n, bt, wt);
    findTurnAroundTime(processes, n, bt, wt, tat);
    cout << "Processes "
         << " Burst time "
         << " Waiting time "
         << " Turn around time\n";
    for (int i = 0; i < n; i++)
    {
        total_wt = total_wt + wt[i];
        total_tat = total_tat + tat[i];
        cout << " " << i + 1 << "\t\t" << bt[i] << "\t "
             << wt[i] << "\t\t " << tat[i] << endl;
    }
    cout << "Average waiting time = "
         << (float)total_wt / (float)n;
    cout << "\nAverage turn around time = "
         << (float)total_tat / (float)n;
}
int main()
{
    int processes[] = {1, 2, 3};
    int n = sizeof processes / sizeof processes[0];
    int burst_time[] = { 1, 2, 8, 6, 4;
    findavgTime(processes, n, burst_time);
    return 0;
}
```

## SJF Code :

```
#include <stdio.h>
#include <vector>
#include <utility>
#include <algorithm>
    using namespace std;
void sjf(vector<pair<int, int>> v)
{
    int waiting_time = 0;
    int total_waiting_time = 0;
    sort(v.begin(), v.end(), [](const pair<int, int> p1, const pair<i nt, int> p2
) -> bool {
        return p1.first < p2.first;
    });
    printf("Waiting time for process %d: %d\n", v[0].second, waiting_t ime);
    for (int i = 0; i < v.size() - 1; i++)
    {
        waiting_time += v[i].first;
        printf("Waiting time for process %d: %d\n", v[i + 1].second, w aiting_tim
e);
        total_waiting_time += waiting_time;
    }
    double avg_waiting_time = (double)total_waiting_time / v.size();
    printf("Average waiting time: %f\n", avg_waiting_time);
}
int main()
{
    int no_of_processes, burst_time;
    vector<pair<int, int>> jobs;
    printf("Enter no of processes: ");
    scanf("%d", &no_of_processes);
    printf("Enter burst time of all the processes: ");
    for (int i = 0; i < no_of_processes; i++)
    {
        scanf("%d", &burst_time);
        jobs.emplace_back(burst_time, i + 1);
    }
    sjf(jobs);
}
```

## SRTF Code :

```
#include <bits/stdc++.h>
    using namespace std;
void srtf(vector<tuple<int, int, int>> v)
{
    sort(v.begin(), v.end(),
         [](tuple<int, int, int> t1, tuple<int, int, int> t2) -> bool {
             return get<1> t1 < get<1> t2;
         });
```

```
}
int main()
{
    int no_of_processes, arrival_time, burst_time;
    vector<tuple<int, int, int>> processes;
    printf("Enter number of processes: ");
    scanf("%d", &no_of_processes);
    for (int i = 0; i < no_of_processes; i++)
    {
        printf("Arrival time for process %d: ", i + 1);
        scanf("%d", &arrival_time);
        printf("Burst time for process %d: ", i + 1);
        scanf("%d", &burst_time);
        printf("\n");
        processes.emplace_back(i + 1, arrival_time, burst_time);
    }
    srtf(processes);
}
```