NAME: MUHAMMAD SAIM NOMANI

ROLL NO :DT-22030

Lab 02: (FCFS, SJF)

1.Implement the First Come First Serve code and paste the output below.

Code :

#include <stdio.h>

void findWaitingTime(int processes[], int n, int bt[], int wt[]) { wt[0] = 0; // Waiting time for the first process is 0 for (int i = 1; i < n; i++) { wt[i] = bt[i - 1] + wt[i - 1]; // Waiting time is sum of previous burst times } }

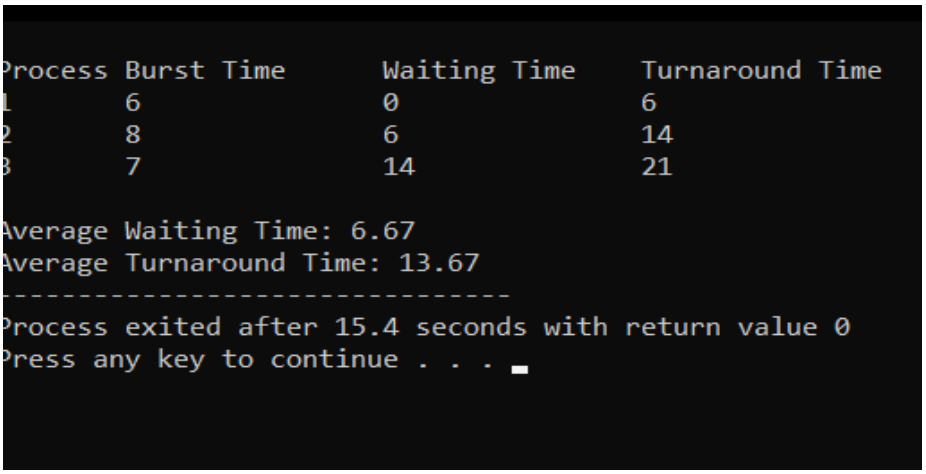
void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[]) { for (int i = 0; i < n; i++) { tat[i] = bt[i] + wt[i]; // Turnaround time is burst time + waiting time } }

void findAvgTime(int processes[], int n, int bt[]) { int wt[n], tat[n]; findWaitingTime(processes, n, bt, wt); findTurnAroundTime(processes, n, bt, wt, tat);

```
int total_wt = 0, total_tat = 0;
printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
for (int i = 0; i < n; i++) {
    total_wt += wt[i];
    total_tat += tat[i];
    printf("%d\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i],
tat[i]);
}

printf("\nAverage Waiting Time: %.2f", (float)total_wt / n);
printf("\nAverage Turnaround Time: %.2f", (float)total_tat / n);


}
```

int main() { int processes[] = {1, 2, 3}; int burst_time[] = {6, 8, 7}; int n = sizeof(processes) / sizeof(processes[0]);

```
findAvgTime(processes, n, burst_time);

return 0;
}
```



2.Implement the Shortest Job First code and paste the output below.

Code :

#include <stdio.h>

void findWaitingTime(int processes[], int n, int bt[], int wt[]) { wt[0] = 0; // Waiting time for the first process is 0 for (int i = 1; i < n; i++) { wt[i] = bt[i - 1] + wt[i - 1]; // Waiting time is sum of previous burst times } }

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]; // Turnaround time is burst time + waiting time } }

void findAvgTime(int processes[], int n, int bt[]) { int wt[n], tat[n]; findWaitingTime(processes, n, bt, wt); findTurnAroundTime(processes, n, bt, wt, tat);

```
int total_wt = 0, total_tat = 0;
printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
for (int i = 0; i < n; i++) {
    total_wt += wt[i];
    total_tat += tat[i];
    printf("%d\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i],
tat[i]);
```

```
}

printf("\nAverage Waiting Time: %.2f", (float)total_wt / n);
printf("\nAverage Turnaround Time: %.2f", (float)total_tat / n);


}
```

void sortByBurstTime(int processes[], int n, int bt[]) { for (int i = 0; i < n - 1; i++) { for (int j = i + 1; j < n; j++) { if (bt[i] > bt[j]) { // Swap burst times int temp = bt[i]; bt[i] = bt[j]; bt[j] = temp;

```
            // Swap corresponding processes
             temp = processes[i];
             processes[i] = processes[j];
             processes[j] = temp;
         }
    }
}


}
```

int main() { int processes[] = {1, 2, 3}; int burst_time[] = {6, 8, 7}; int n = sizeof(processes) / sizeof(processes[0]);

```
// Sort processes by burst time
sortByBurstTime(processes, n, burst_time);

findAvgTime(processes, n, burst_time);

return 0;}
```

```
Process Burst Time      Waiting Time    Turnaround Time
1       6               0               6
3       7               6               13
2       8               13              21

Average Waiting Time: 6.33
Average Turnaround Time: 13.33
--------------------------------
Process exited after 18.33 seconds with return value 0
Press any key to continue . . .
```