Lab Practical 8 Submission

Roll No. and Name: 21BEI036 Meet Patel

Course Code and Name: 2CSOE53 OS

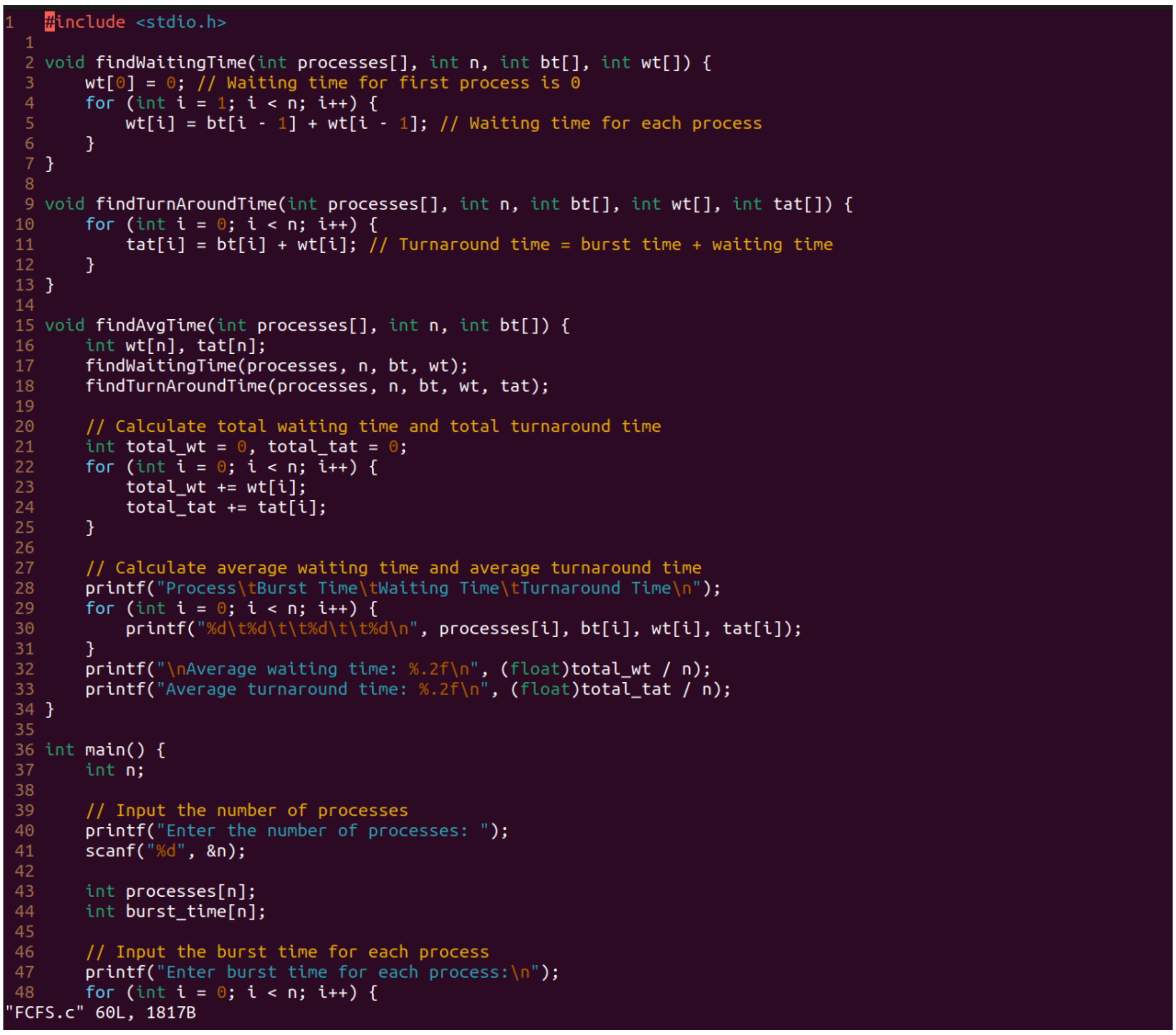
Aim:

a) Write a C program to implement FCFS.

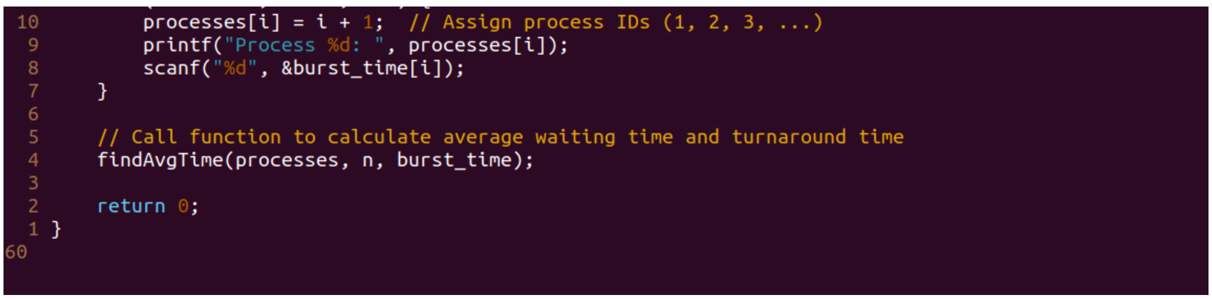
b) Write a a C program to implement Round Robin.

code:

a)

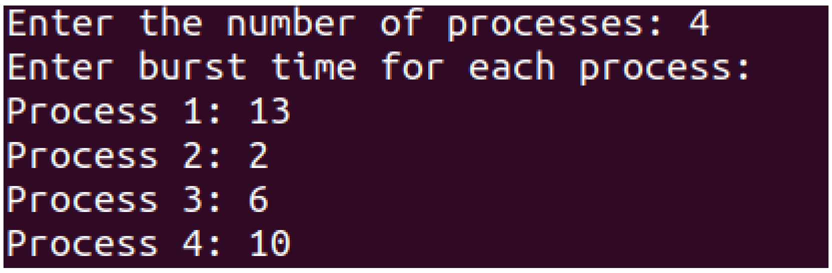


```c
#include <stdio.h>

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

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 = 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);

    // Calculate total waiting time and total turnaround time
    int total_wt = 0, total_tat = 0;
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
    }

    // Calculate average waiting time and average turnaround time
    printf("Process\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (int i = 0; i < n; 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\n", (float)total_wt / n);
    printf("Average turnaround time: %.2f\n", (float)total_tat / n);
}

int main() {
    int n;

    // Input the number of processes
    printf("Enter the number of processes: ");
    scanf("%d", &n);

    int processes[n];
    int burst_time[n];

    // Input the burst time for each process
    printf("Enter burst time for each process:\n");
    for (int i = 0; i < n; i++) {
"FCFS.c" 60L, 1817B
```



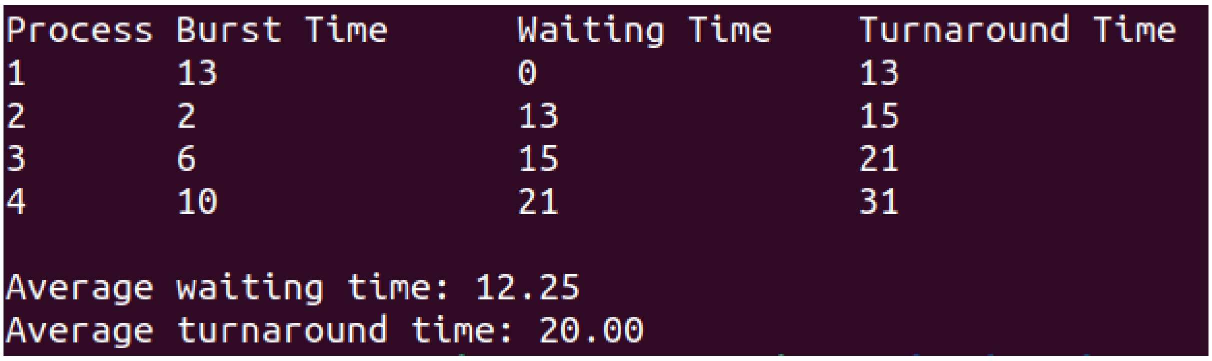
```c
        processes[i] = i + 1;  // Assign process IDs (1, 2, 3, ...)
        printf("Process %d: ", processes[i]);
        scanf("%d", &burst_time[i]);
    }

    // Call function to calculate average waiting time and turnaround time
    findAvgTime(processes, n, burst_time);

    return 0;
}
```

input:



```
Enter the number of processes: 4
Enter burst time for each process:
Process 1: 13
Process 2: 2
Process 3: 6
Process 4: 10
```

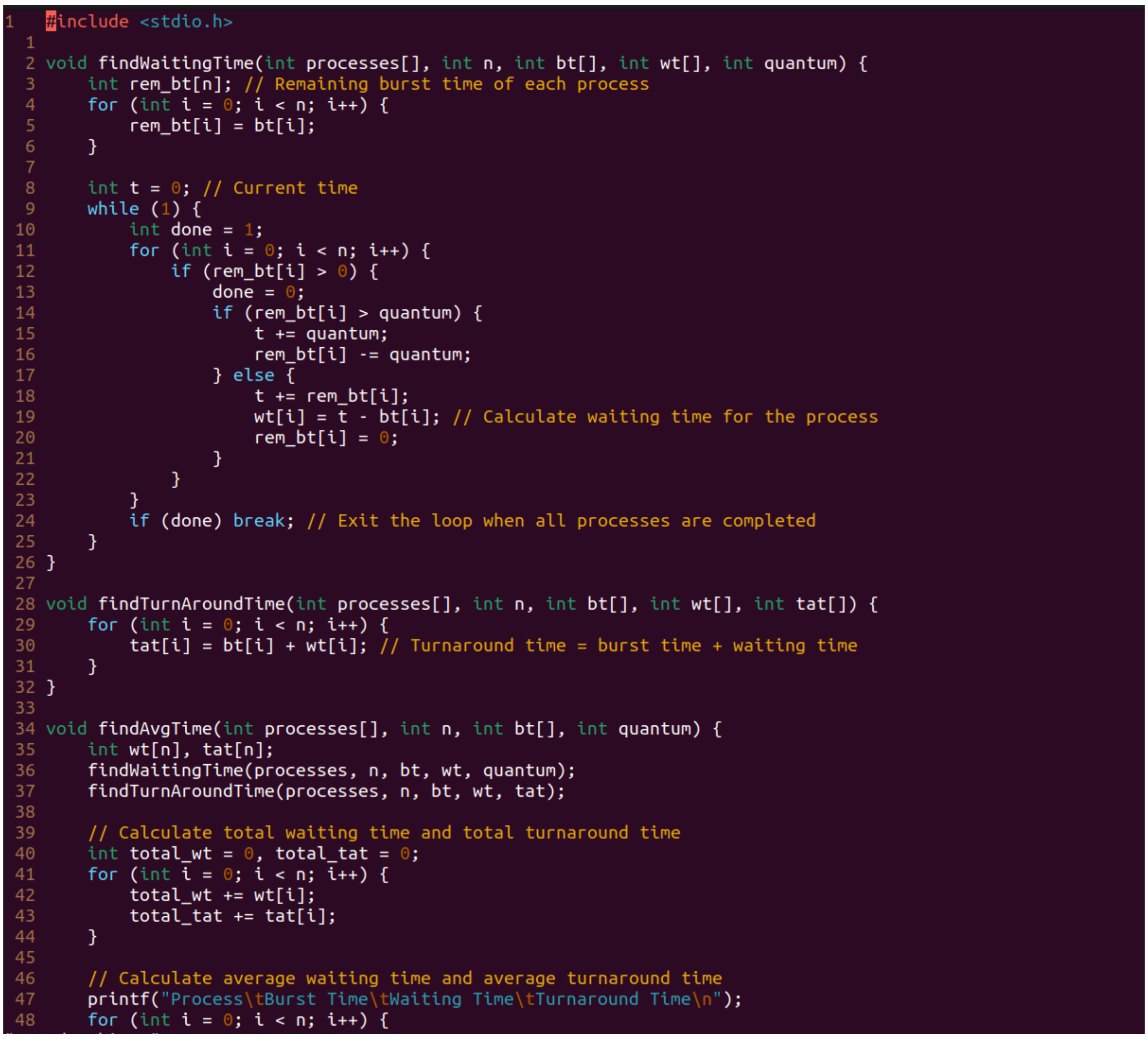
output:



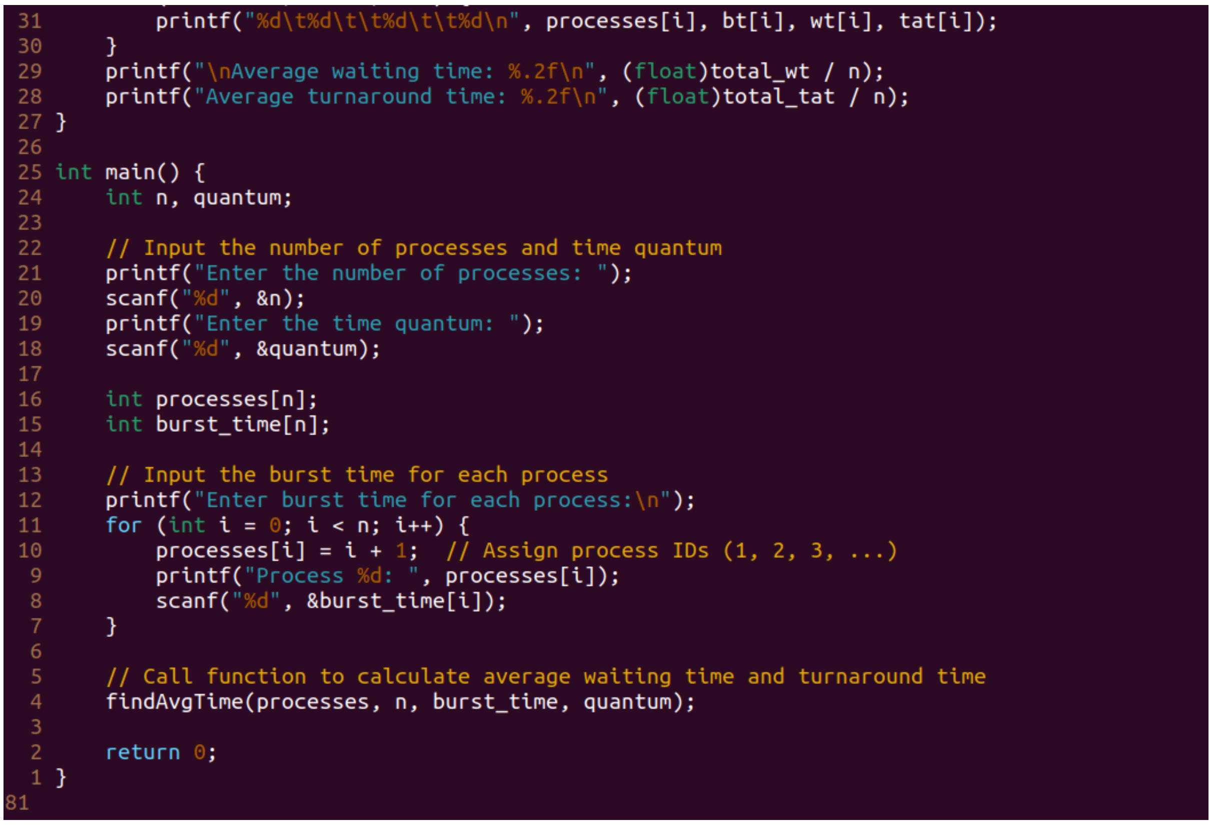
```
Process Burst Time      Waiting Time    Turnaround Time
1       13              0               13
2       2               13              15
3       6               15              21
4       10              21              31

Average waiting time: 12.25
Average turnaround time: 20.00
```

b)

<u>Code:</u>

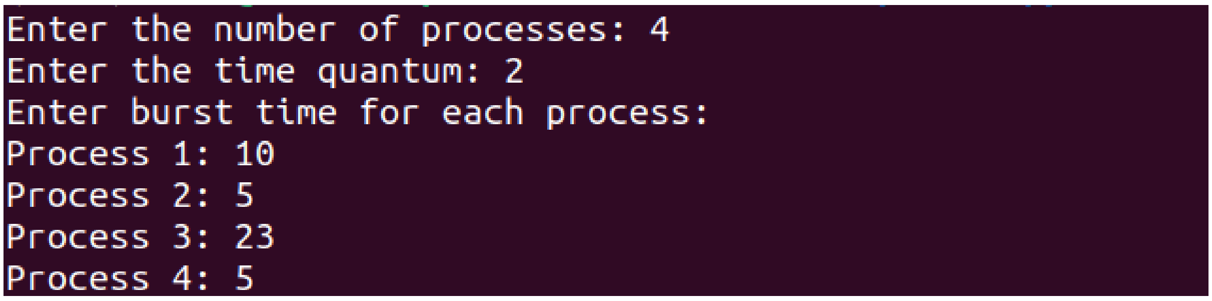


```c
#include <stdio.h>

void findWaitingTime(int processes[], int n, int bt[], int wt[], int quantum) {
    int rem_bt[n]; // Remaining burst time of each process
    for (int i = 0; i < n; i++) {
        rem_bt[i] = bt[i];
    }

    int t = 0; // Current time
    while (1) {
        int done = 1;
        for (int i = 0; i < n; i++) {
            if (rem_bt[i] > 0) {
                done = 0;
                if (rem_bt[i] > quantum) {
                    t += quantum;
                    rem_bt[i] -= quantum;
                } else {
                    t += rem_bt[i];
                    wt[i] = t - bt[i]; // Calculate waiting time for the process
                    rem_bt[i] = 0;
                }
            }
        }
        if (done) break; // Exit the loop when all processes are completed
    }
}

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 = burst time + waiting time
    }
}

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

    // Calculate total waiting time and total turnaround time
    int total_wt = 0, total_tat = 0;
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
    }

    // Calculate average waiting time and average turnaround time
    printf("Process\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (int i = 0; i < n; i++) {
```



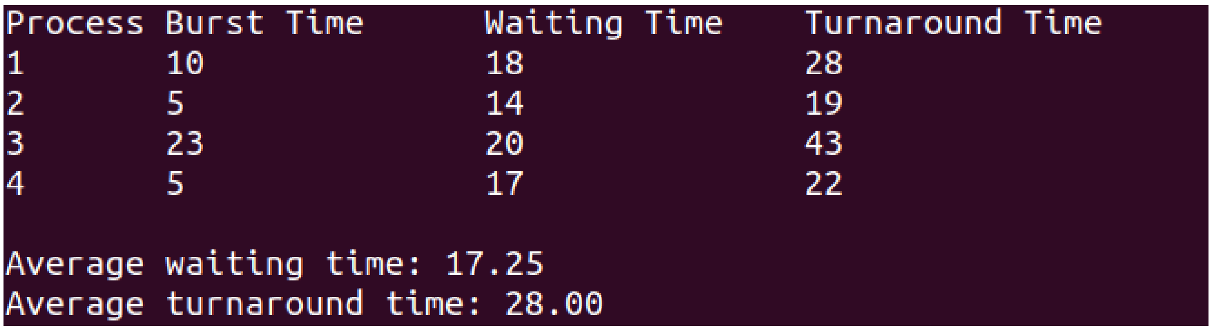
```c
        printf("%d\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i], tat[i]);
    }
    printf("\nAverage waiting time: %.2f\n", (float)total_wt / n);
    printf("Average turnaround time: %.2f\n", (float)total_tat / n);
}

int main() {
    int n, quantum;

    // Input the number of processes and time quantum
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    printf("Enter the time quantum: ");
    scanf("%d", &quantum);

    int processes[n];
    int burst_time[n];

    // Input the burst time for each process
    printf("Enter burst time for each process:\n");
    for (int i = 0; i < n; i++) {
        processes[i] = i + 1;  // Assign process IDs (1, 2, 3, ...)
        printf("Process %d: ", processes[i]);
        scanf("%d", &burst_time[i]);
    }

    // Call function to calculate average waiting time and turnaround time
    findAvgTime(processes, n, burst_time, quantum);

    return 0;
}
```

input:



```
Enter the number of processes: 4
Enter the time quantum: 2
Enter burst time for each process:
Process 1: 10
Process 2: 5
Process 3: 23
Process 4: 5
```

output:



```
Process Burst Time      Waiting Time     Turnaround Time
1       10              18               28
2       5               14               19
3       23              20               43
4       5               17               22

Average waiting time: 17.25
Average turnaround time: 28.00
```