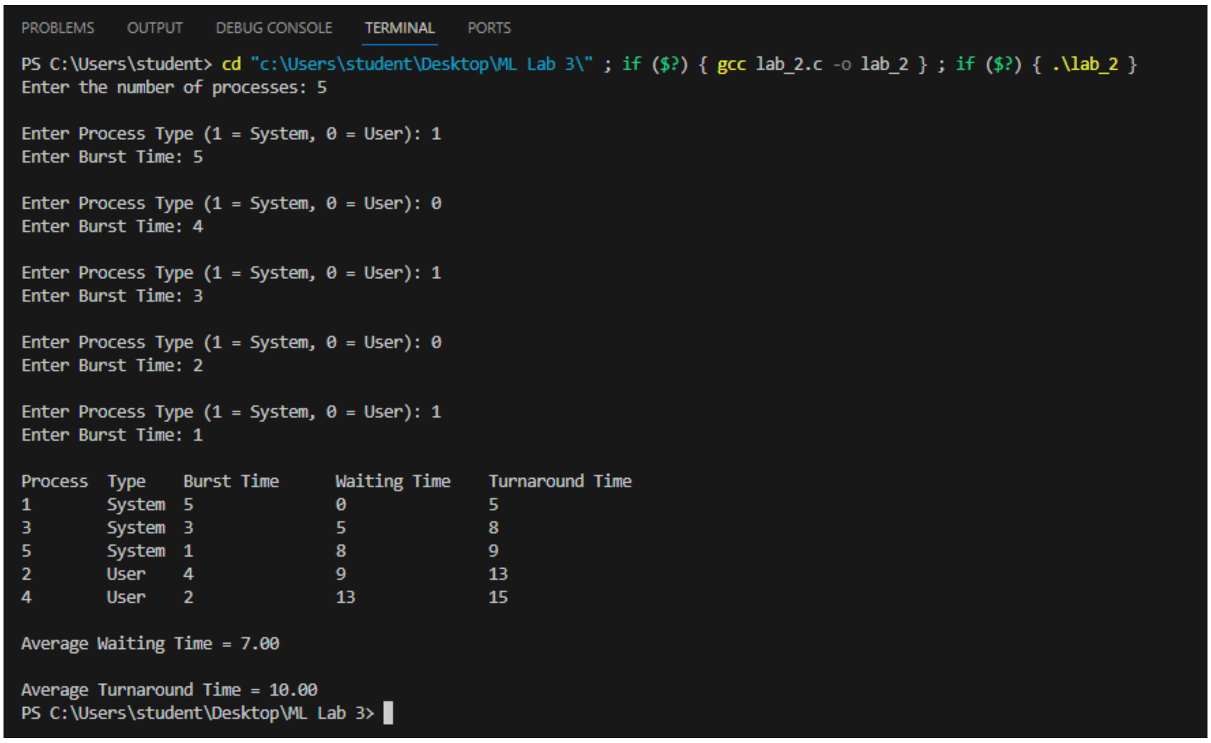
LAB-2

2. Write a C program to simulate a multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided into two categories:
   system processes and user processes. System processes are to be given higher priority than user processes. Use FCFS scheduling for the processes in each queue.

```c
#include<stdio.h>

typedef struct{
    int id;
    int burst_time;
    int is_system;
} Process;

void calculate_scheduling(Process processes[], int n) {
    int waiting_time[n], turnaround_time[n];
    float total_wt = 0, total_tat = 0;

    waiting_time[0] = 0;
    turnaround_time[0] = processes[0].burst_time;

    total_wt = waiting_time[0];
    total_tat = turnaround_time[0];

    for (int i = 1; i < n; i++) {
        waiting_time[i] = waiting_time[i - 1] + processes[i - 1].burst_time;
        turnaround_time[i] = waiting_time[i] + processes[i].burst_time;
        total_wt += waiting_time[i];
        total_tat += turnaround_time[i];
    }

    printf("\nProcess\t Type\t Burst Time\t Waiting Time\t Turnaround
Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t %s\t %d\t\t %d\t\t %d\n",
               processes[i].id,
               processes[i].is_system ? "System" : "User",
               processes[i].burst_time,
               waiting_time[i],
               turnaround_time[i]);
    }

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

int main() {
    int n, sys_count = 0, user_count = 0;

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

    if (n <= 0) {
        printf("No processes to schedule.\n");
        return 0;
    }

    Process system_queue[n], user_queue[n];

    for (int i = 0; i < n; i++) {
        int type, burst;
        printf("\nEnter Process Type (1 = System, 0 = User): ");
        scanf("%d", &type);
        printf("Enter Burst Time: ");
        scanf("%d", &burst);

        Process p = {i + 1, burst, type};

        if (type == 1)
            system_queue[sys_count++] = p;
        else
            user_queue[user_count++] = p;
    }

    Process processes[n];
    int index=0;

    for (int i = 0; i < sys_count; i++)
        processes[index++] = system_queue[i];

    for (int i = 0; i < user_count; i++)
        processes[index++] = user_queue[i];

    calculate_scheduling(processes, n);

    return 0;
}
```

OUTPUT:



```
PS C:\Users\student> cd "c:\Users\student\Desktop\ML Lab 3\" ; if ($?) { gcc lab_2.c -o lab_2 } ; if ($?) { .\lab_2 }
Enter the number of processes: 5

Enter Process Type (1 = System, 0 = User): 1
Enter Burst Time: 5

Enter Process Type (1 = System, 0 = User): 0
Enter Burst Time: 4

Enter Process Type (1 = System, 0 = User): 1
Enter Burst Time: 3

Enter Process Type (1 = System, 0 = User): 0
Enter Burst Time: 2

Enter Process Type (1 = System, 0 = User): 1
Enter Burst Time: 1

Process  Type    Burst Time      Waiting Time    Turnaround Time
1        System  5               0               5
3        System  3               5               8
5        System  1               8               9
2        User    4               9               13
4        User    2               13              15

Average Waiting Time = 7.00

Average Turnaround Time = 10.00
PS C:\Users\student\Desktop\ML Lab 3> 
```