ASSIGNMENT-3 (SCHEDULING)

Name: Monika Raghav

Roll.No.: 2301420010

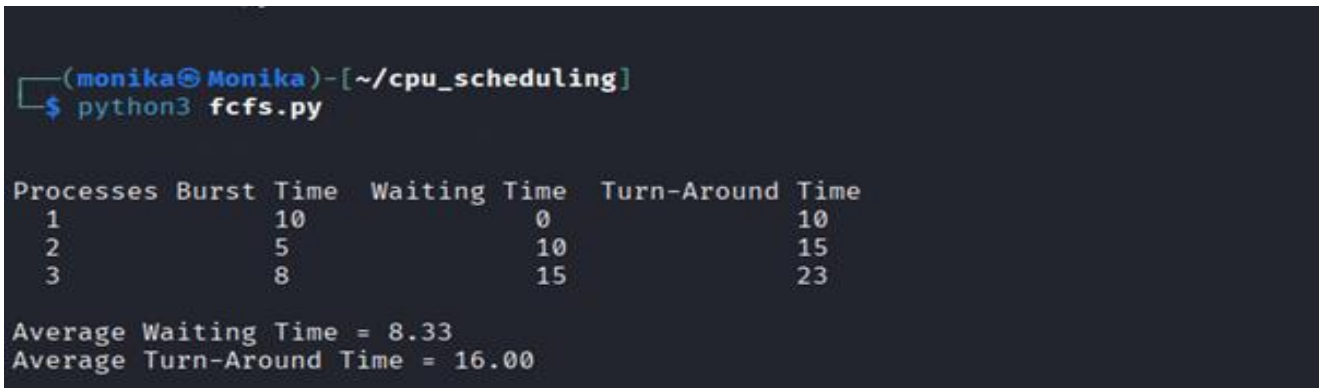
Course: B.Tech(CSE) Data Science

1. First Come First Serve (FCFS) :

Code:

```
File  Actions  Edit  View  Help
 GNU nano 8.4                                    fcfs.py
# First Come First Serve (FCFS) CPU Scheduling

def findWaitingTime(processes, n, bt, wt):
    wt[0] = 0
    for i in range(1, n):
        wt[i] = bt[i - 1] + wt[i - 1]

def findTurnAroundTime(processes, n, bt, wt, tat):
    for i in range(n):
        tat[i] = bt[i] + wt[i]

def findavgTime(processes, n, bt):
    wt = [0] * n
    tat = [0] * n

    findWaitingTime(processes, n, bt, wt)
    findTurnAroundTime(processes, n, bt, wt, tat)

    total_wt = sum(wt)
    total_tat = sum(tat)
    print("\nProcesses Burst Time  Waiting Time  Turn-Around Time")

    for i in range(n):
        print(f"  {processes[i]}\t\t{bt[i]}\t\t{wt[i]}\t\t{tat[i]}")

    print(f"\nAverage Waiting Time = {total_wt / n:.2f}")
    print(f"Average Turn-Around Time = {total_tat / n:.2f}")

if __name__ == "__main__":
    processes = [1, 2, 3]
    burst_time = [10, 5, 8]
    findavgTime(processes, len(processes), burst_time)
```

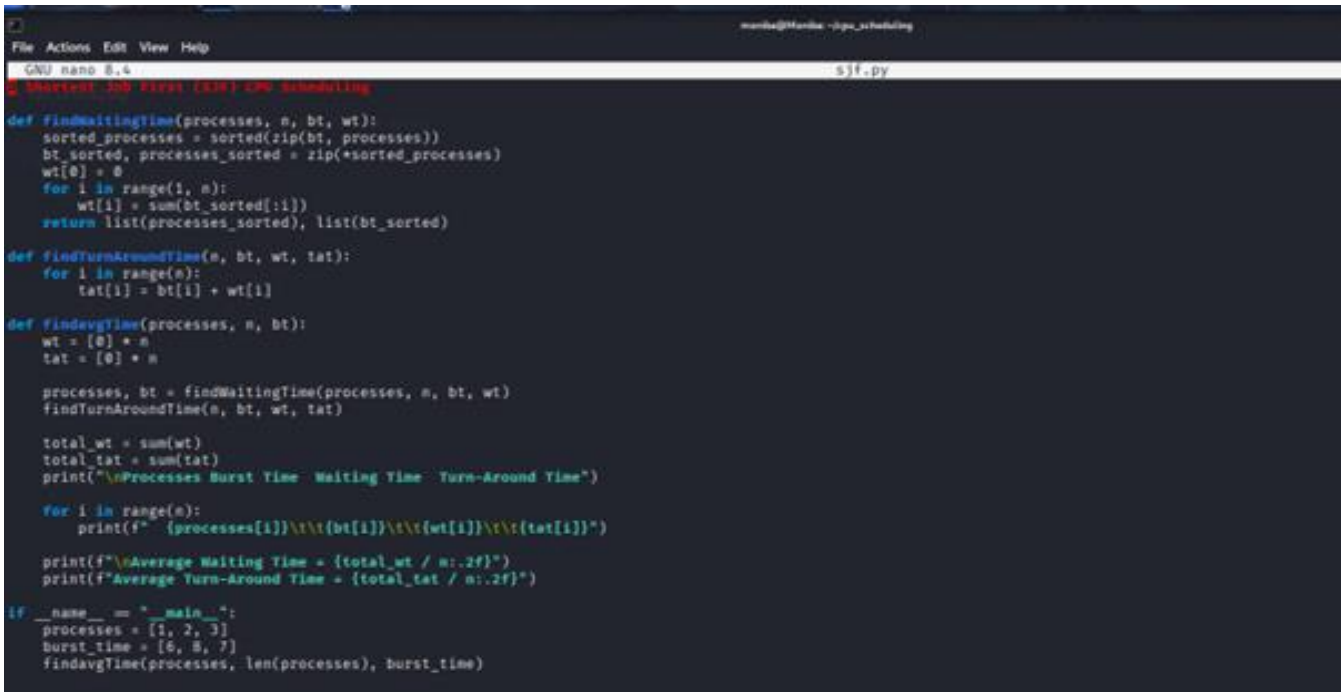
Output:



```
┌──(monika㉿Monika)-[~/cpu_scheduling]
└─$ python3 fcfs.py

Processes Burst Time  Waiting Time  Turn-Around Time
  1             10              0               10
  2             5               10              15
  3             8               15              23

Average Waiting Time = 8.33
Average Turn-Around Time = 16.00
```

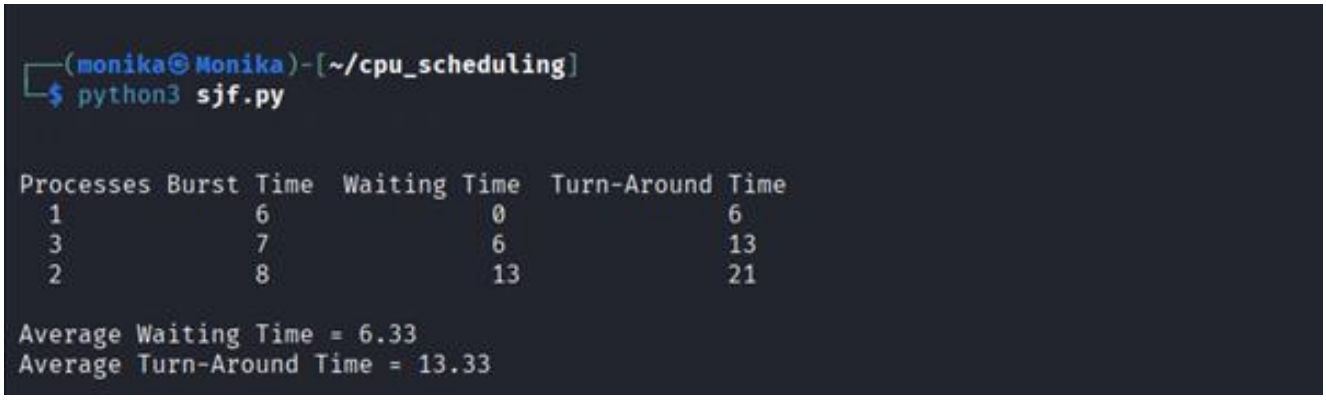
2. Shortest Job First (SJF) :

Code:



```
File  Actions  Edit  View  Help
 GNU nano 8.4                                    sjf.py
# Shortest Job First (SJF) CPU Scheduling

def findWaitingTime(processes, n, bt, wt):
    sorted_processes = sorted(zip(bt, processes))
    bt_sorted, processes_sorted = zip(*sorted_processes)
    wt[0] = 0
    for i in range(1, n):
        wt[i] = sum(bt_sorted[:i])
    return list(processes_sorted), list(bt_sorted)

def findTurnAroundTime(n, bt, wt, tat):
    for i in range(n):
        tat[i] = bt[i] + wt[i]

def findavgTime(processes, n, bt):
    wt = [0] * n
    tat = [0] * n

    processes, bt = findWaitingTime(processes, n, bt, wt)
    findTurnAroundTime(n, bt, wt, tat)

    total_wt = sum(wt)
    total_tat = sum(tat)
    print("\nProcesses Burst Time  Waiting Time  Turn-Around Time")

    for i in range(n):
        print(f"  {processes[i]}\t\t{bt[i]}\t\t{wt[i]}\t\t{tat[i]}")

    print(f"\nAverage Waiting Time = {total_wt / n:.2f}")
    print(f"Average Turn-Around Time = {total_tat / n:.2f}")

if __name__ == "__main__":
    processes = [1, 2, 3]
    burst_time = [6, 8, 7]
    findavgTime(processes, len(processes), burst_time)
```

Output:



```
┌──(monika㉿Monika)-[~/cpu_scheduling]
└─$ python3 sjf.py

Processes Burst Time  Waiting Time  Turn-Around Time
  1             6               0               6
  3             7               6               13
  2             8               13              21

Average Waiting Time = 6.33
Average Turn-Around Time = 13.33
```

3. Round Robin:

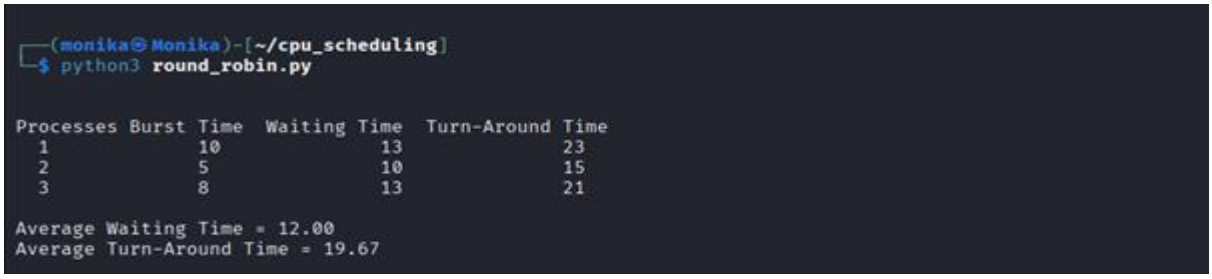
Code:

```
File  Actions  Edit  View  Help
 GNU nano 8.4                                    round_robin.py *
# Round Robin (RR) CPU Scheduling

def findWaitingTime(processes, n, bt, wt, quantum):
    rem_bt = bt[:]
    t = 0
    while True:
        done = True
        for i in range(n):
            if rem_bt[i] > 0:
                done = False
                if rem_bt[i] > quantum:
                    t += quantum
                    rem_bt[i] -= quantum
                else:
                    t += rem_bt[i]
                    wt[i] = t - bt[i]
                    rem_bt[i] = 0
        if done:
            break

def findTurnAroundTime(n, bt, wt, tat):
    for i in range(n):
        tat[i] = bt[i] + wt[i]

def findavgTime(processes, n, bt, quantum):
    wt = [0] * n
    tat = [0] * n

    findWaitingTime(processes, n, bt, wt, quantum)
    findTurnAroundTime(n, bt, wt, tat)

    total_wt = sum(wt)
    total_tat = sum(tat)
    print("\nProcesses Burst Time  Waiting Time  Turn-Around Time")

    for i in range(n):
        print(f"  {processes[i]}\t\t{bt[i]}\t\t{wt[i]}\t\t{tat[i]}")
```



```
    print(f"\nAverage Waiting Time = {total_wt / n:.2f}")
    print(f"Average Turn-Around Time = {total_tat / n:.2f}")

if __name__ == "__main__":
    processes = [1, 2, 3]
    burst_time = [10, 5, 8]
    quantum = 2
    findavgTime(processes, len(processes), burst_time, quantum)
```

Output:



```
┌──(monika㉿Monika)-[~/cpu_scheduling]
└─$ python3 round_robin.py

Processes Burst Time  Waiting Time  Turn-Around Time
  1             10              13              23
  2             5               10              15
  3             8               13              21

Average Waiting Time = 12.00
Average Turn-Around Time = 19.67
```

4. Priority Schedule :

Code:

```
# ------------------------------------------------
# Program: Priority Scheduling (Non-Preemptive)
# Subject: Operating System
# Language: Python
# ------------------------------------------------

# Define a class to represent each process
class Process:
    def __init__(self, pid, bt, pri):
        self.pid = pid          # Process ID
        self.bt = bt            # Burst Time
        self.pri = pri          # Priority (lower number = higher priority)
        self.wt = 0             # Waiting Time
        self.tat = 0            # Turnaround Time

# Step 1: Input number of processes
n = int(input("Enter number of processes: "))

processes = []

# Step 2: Input Burst Time and Priority for each process
for i in range(n):
    bt, pri = map(int, input(f"Enter Burst Time and Priority for Process {i+1}: ").split())
    processes.append(Process(i + 1, bt, pri))

# Step 3: Sort processes by priority (ascending)
processes.sort(key=lambda x: x.pri)

# Step 4: Calculate Waiting Time and Turnaround Time
processes[0].wt = 0
processes[0].tat = processes[0].bt

for i in range(1, n):
    processes[i].wt = processes[i-1].wt + processes[i-1].bt
    processes[i].tat = processes[i].wt + processes[i].bt

# Step 5: Display process table
print("\n-------------------------------------------------")
print("Process\tBT\tPriority\tWT\tTAT")
print("-------------------------------------------------")

total_wt = 0
total_tat = 0
```

```
for p in processes:
    print(f"P{p.pid}\t{p.bt}\t{p.pri}\t\t{p.wt}\t{p.tat}")
    total_wt += p.wt
    total_tat += p.tat

# Step 6: Display averages
avg_wt = total_wt / n
avg_tat = total_tat / n

print("-------------------------------------------------")
print(f"Average Waiting Time: {avg_wt:.2f}")
print(f"Average Turnaround Time: {avg_tat:.2f}")
```

Output:

```
┌──(monika㉿Monika)-[~]
└─$ python3 priority_scheduling.py

Enter number of processes: 4
Enter Burst Time and Priority for Process 1: 6 2
Enter Burst Time and Priority for Process 2: 8 1
Enter Burst Time and Priority for Process 3: 7 3
Enter Burst Time and Priority for Process 4: 3 4

-------------------------------------------------
Process BT      Priority        WT      TAT
-------------------------------------------------
P2      8       1               0       8
P1      6       2               8       14
P3      7       3               14      21
P4      3       4               21      24
-------------------------------------------------
Average Waiting Time: 10.75
Average Turnaround Time: 16.75
```