Ex. No. : 6a **Date : 21.02.2025**

Register No. : 230701385 **Name : VISHWAK S**

FIRST COME FIRST SERVE

Aim:

To implement First-come First- serve (FCFS) scheduling technique.

Algorithm:

1. Get the number of processes from the user.
2. Read the process name and burst time.
3. Calculate the total process time.
4. Calculate the total waiting time and total turnaround time for each process.
5. Display the process name & burst time for each process.
6. Display the total waiting time, average waiting time, turnaround time.

Program:

D: > OS > FCFS.py > n

```
n=int(input("Enter the number of process"))
bt=[]
tat=[]
wt=[]
for i in range(1,n+1):
    print("Enter the burst time for process",i)
    t=int(input())
    bt.append(t)
print(bt)
tat.append(bt[0])
for i in range(2,n+1):
      t=bt[i-1]+tat[i-2]
      tat.append(t)
print("Proces\t\tBurst Time\tTurn Around Time\tWaiting Time")
for i in range(0,n):
    t=tat[i]-bt[i]
    wt.append(t)
    print("\n",i,"\t\t",bt[i],"\t\t\t",tat[i],"\t\t\t",wt[i])
sum=0
sum1=0
for i in range(0,n):
     sum+=tat[i]
     sum1+=wt[i]

print("Average Turn Around Time :",sum/n)
print("Average Waiting Time :",sum1/n)
```

Output:

```
PS C:\Users\sudha> & C:/Users/sudha/AppData/Local/Programs/Python/Python313/python.exe d:/OS/FCFS.py
Enter the number of process3
Enter the burst time for process 1
5
Enter the burst time for process 2
3
Enter the burst time for process 3
8
[5, 3, 8]
Proces          Burst Time      Turn Around Time        Waiting Time

 0              5                       5                       0

 1              3                       8                       5

 2              8                       16                      8
Average Turn Around Time : 9.666666666666666
Average Waiting Time : 4.333333333333333
PS C:\Users\sudha>
```

Result:

Hence the implementation of First-come First- serve (FCFS) scheduling technique has been executed successfully.

Ex. No. : 6b **Date : 21.02.2025**

Register No. : 230701385 **Name : VISHWAK S**

SHORTEST JOB FIRST

Aim:

To implement the Shortest Job First (SJF) scheduling technique

Algorithm:

1. Declare the structure and its elements.

2. Get a number of processes as input from the user.

3. Read the process name, arrival time and burst time

4. Initialize waiting time, turnaround time & flag of read processes to zero.

5. Sort based on burst time of all processes in ascending order

6. Calculate the waiting time and turnaround time for each process.

7. Calculate the average waiting time and average turnaround time.

8. Display the results.

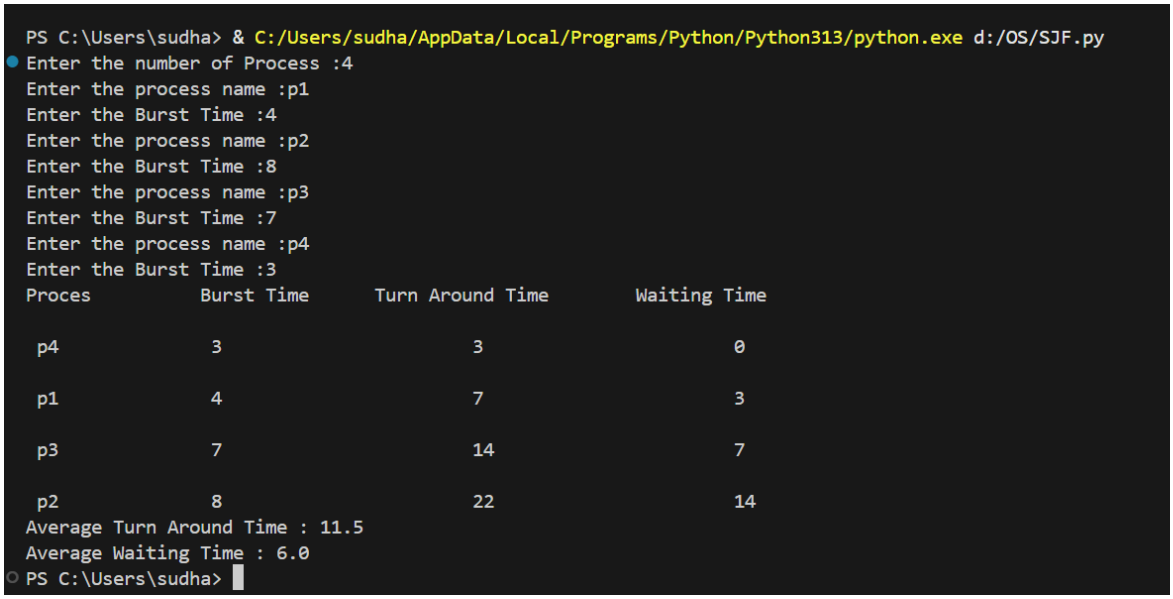
Program:

D: > OS > SJF.py > ...

```python
from operator import itemgetter
n=int(input("Enter the number of Process :"))
sjf=[]
tat=[]
for i in range(0,n):
    p_name=input("Enter the process name :")
    bt=int(input("Enter the Burst Time :"))
    sjf.append([p_name,bt])
sjf = sorted(sjf, key=itemgetter(1))
tat.append(sjf[0][1])
for i in range(1,n):
    t=tat[i-1]+sjf[i][1]
    tat.append(t)
wt=[]
for i in range(0,n):
    t=tat[i]-sjf[i][1]
    wt.append(t)
print("Proces\t\tBurst Time\tTurn Around Time\tWaiting Time")
for i in range(0,n):
    print("\n",sjf[i][0],"\t\t",sjf[i][1],"\t\t\t",tat[i],"\t\t\t",wt[i])
sum=0
sum1=0
for i in range(0,n):
    sum+=tat[i]
    sum1+=wt[i]

print("Average Turn Around Time :",sum/n)
print("Average Waiting Time :",sum1/n)
```

Output:



Result:

Hence the implementation of Shortest Job First (SJF) scheduling technique has been executed successfully.

Ex. No. : 6c **Date : 28.02.2025**

Register No. : 230701385 **Name : VISHWAK S**

PRIORITY SCHEDULING

Aim:

To implement priority scheduling technique.

Algorithm:

1. Get the number of processes from the user.
2. Read the process name, burst time and priority of the process.
3. Sort based on burst time of all processes in ascending order based priority.
4. Calculate the total waiting time and total turnaround time for each pr
5. Display the process name & burst time for each process.
6. Display the total waiting time, average waiting time, turnaround time.

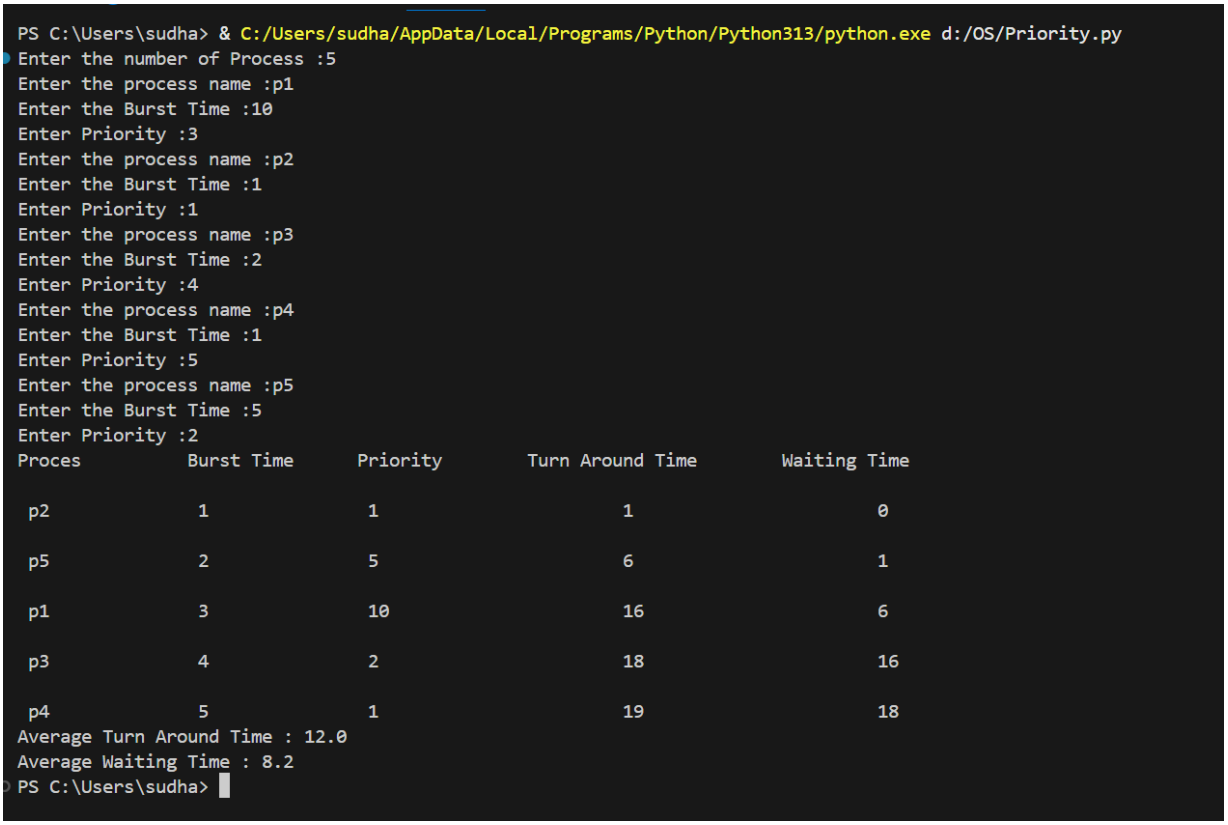
Program:

D: > OS > Priority.py > ...

```python
from operator import itemgetter
n=int(input("Enter the number of Process :"))
list1=[]
tat=[]
for i in range(0,n):
    p_name=input("Enter the process name :")
    bt=int(input("Enter the Burst Time :"))
    prior=int(input("Enter Priority :"))
    list1.append([p_name,bt,prior])
list1 = sorted(list1, key=itemgetter(2))
tat.append(list1[0][1])
for i in range(1,n):
    t=tat[i-1]+list1[i][1]
    tat.append(t)
wt=[]
for i in range(0,n):
    t=tat[i]-list1[i][1]
    wt.append(t)
print("Proces\t\tBurst Time\tPriority\tTurn Around Time\tWaiting Time")
for i in range(0,n):
    print("\n",list1[i][0],"\t\t",list1[i][2],"\t\t",list1[i][1],"\t\t\t",tat[i],"\t\t\t",wt[i])
sum=0
sum1=0
for i in range(0,n):
    sum+=tat[i]
    sum1+=wt[i]

print("Average Turn Around Time :",sum/n)
print("Average Waiting Time :",sum1/n)
```

Output:



```
PS C:\Users\sudha> & C:/Users/sudha/AppData/Local/Programs/Python/Python313/python.exe d:/OS/Priority.py
Enter the number of Process :5
Enter the process name :p1
Enter the Burst Time :10
Enter Priority :3
Enter the process name :p2
Enter the Burst Time :1
Enter Priority :1
Enter the process name :p3
Enter the Burst Time :2
Enter Priority :4
Enter the process name :p4
Enter the Burst Time :1
Enter Priority :5
Enter the process name :p5
Enter the Burst Time :5
Enter Priority :2
Proces          Burst Time      Priority        Turn Around Time        Waiting Time

 p2             1               1                       1                       0

 p5             2               5                       6                       1

 p1             3               10                      16                      6

 p3             4               2                       18                      16

 p4             5               1                       19                      18
Average Turn Around Time : 12.0
Average Waiting Time : 8.2
PS C:\Users\sudha>
```

Result:

Hence the implementation of Priority scheduling technique has been executed successfully.

Ex. No. : 6d **Date : 28.02.2025**

Register No. : 230701385 **Name : VISHWAK S**

ROUND ROBIN SCHEDULING

Aim:

To implement the Round Robin (RR) scheduling technique

Algorithm:

1. Declare the structure and its elements.

2. Get number of processes and Time quantum as input from the user.

3. Read the process name, arrival time and burst time

4. Create an array **rem_bt[]** to keep track of remaining burst time of processes which is initially copy of bt[] (burst times array)

5. Create another array **wt[]** to store waiting times of processes. Initialize this array as 0.

6. Initialize time : t = 0

7. Keep traversing all processes while all processes are not done.
   Do the following for i'th process if it is not done yet.

   a- If rem_bt[i] > quantum

   (i) t = t + quantum

   (ii) bt_rem[i] -= quantum;

   b- Else // Last cycle for this process

   (i) t = t + bt_rem[i];

   (ii) wt[i] = t - bt[i]

   (iii) bt_rem[i] = 0; // This process is over

8. Calculate the waiting time and turnaround time for each process.

9. Calculate the average waiting time and average turnaround time.

10. Display the results.

Program:

D: > OS > Round_robin.py > n

```python
n = int(input("Enter number of processes: "))
process = []
for i in range(0,n):
    process.append(i+1)
bt=[]
for i in range(0,n):
    burst_time = int(input("Enter burst time of process: "))
    bt.append(burst_time)
tq = int(input("Enter time quantum: "))
wt = [0]*n
tat = [0]*n
rem_bt=bt.copy()
time = 0
while(1):
    complete = True
    for i in range(n):
        if rem_bt[i] > 0:
            complete = False
            if rem_bt[i] > tq:
                time += tq
                rem_bt[i] -= tq
            else:
                time += rem_bt[i]
                wt[i] = time - bt[i]
                rem_bt[i] = 0
    if complete:
        break
for i in range(0,n):
    tat[i] = bt[i] + wt[i]
total_wt = 0
total_tat = 0
for i in range(0,n):
    total_wt = total_wt + wt[i]
    total_tat = total_tat + tat[i]
print("Processes    Burst Time     WaitingTime    Turn-Around Time")
for i in range(0,n):
    print( i + 1, "\t\t", bt[i],"\t\t", wt[i], "\t\t", tat[i])
print("Average turn around time = ",(total_tat / n))
print("\nAverage waiting time = ",(total_wt / n))

```

Output:

```
PS C:\Users\sudha> & C:/Users/sudha/AppData/Local/Programs/Python/Python313/python.exe d:/OS/Round_robin.py
Enter number of processes: 3
Enter burst time of process: 4
Enter burst time of process: 5
Enter burst time of process: 3
Enter time quantum: 2
Processes    Burst Time     WaitingTime    Turn-Around Time
1               4              4              8
2               5              7              12
3               3              8              11
Average turn around time =  10.333333333333334

Average waiting time =  6.333333333333333
PS C:\Users\sudha>
```

Result:

Hence the implementation of the Round Robin scheduling technique has been executed successfully.