**SVKM’S NMIMS**

**MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT& ENGINEERING**

**(Campus Name)**

Academic Year: 2020-2021

# **Practical 3- Shortest Job first scheduling algorithm**

|  |  |  |
| --- | --- | --- |
| Name | Roll\_No | SAP-ID |
| Jal Bafana | K005 | 70102300054 |

Dear all,

Kindly complete the following task with your name in output file also attach the C program with the file.

Find the Turnaround time and Average Turnaround time.

Find the Waiting time and Average Waiting time.

1. Completion Time: Time at which process completes its execution.
2. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
3. Waiting Time(W.T): Time Difference between turn around time and burst time.   
   Waiting Time = Turn Around Time – Burst Time

**CODE:**

#include<stdio.h>

int main()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

for(i=0;i<n;i++)

{

printf("P%d:",i+1);

scanf("%d",&bt[i]);

p[i]=i+1;

}

//sorting of burst times

for(i=0;i<n;i++)

{

pos=i;

for(j=i+1;j<n;j++)

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=(float)total/n;

total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

for(i=0;i<n;i++)

{

tat[i]=bt[i]+wt[i];

total+=tat[i];

printf("\nP%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

avg\_tat=(float)total/n;

printf("\n\nAverage Waiting Time=%f",avg\_wt);

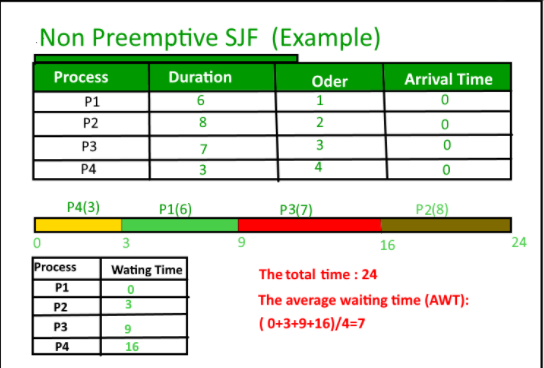
printf("\n\nAverage Turnaround Time=%f\n",avg\_tat);

printf("\n\n Name\n");

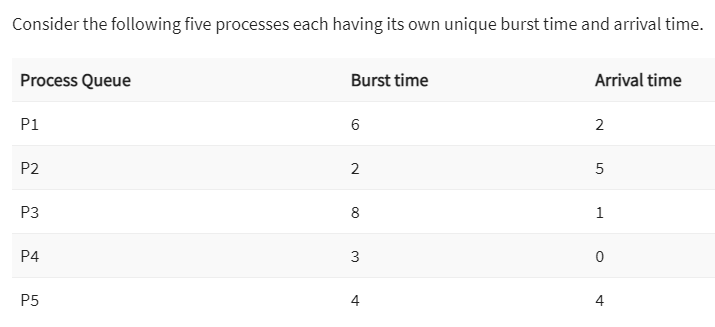
}

**OUTPUT:**

### **Example 1:-**



**Example 2:-**



def sjf\_scheduling():

n = int(input("Enter number of processes: "))

bt = [] # Burst time list

p = [] # Process list

wt = [] # Waiting time list

tat = [] # Turnaround time list

total = 0

# Input burst time for each process

print("\nEnter Burst Time:")

for i in range(n):

burst\_time = int(input(f"P{i+1}: "))

bt.append(burst\_time)

p.append(i+1)

# Sorting the burst time with its corresponding process number

for i in range(n):

for j in range(i+1, n):

if bt[j] < bt[i]:

bt[i], bt[j] = bt[j], bt[i]

p[i], p[j] = p[j], p[i]

# Waiting time calculation

wt.append(0) # First process has no waiting time

for i in range(1, n):

wt.append(sum(bt[:i])) # Waiting time for process i

total\_wt = sum(wt)

avg\_wt = total\_wt / n

# Turnaround time calculation

for i in range(n):

tat.append(bt[i] + wt[i]) # Turnaround time = Burst time + Waiting time

total\_tat = sum(tat)

avg\_tat = total\_tat / n

# Output the results

print("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time")

for i in range(n):

print(f"P{p[i]}\t\t{bt[i]}\t\t{wt[i]}\t\t{tat[i]}")

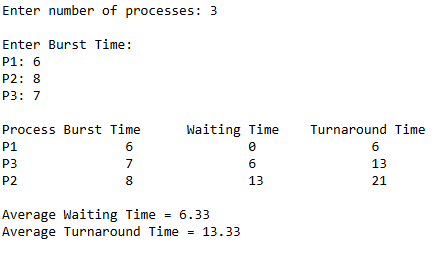
print(f"\nAverage Waiting Time = {avg\_wt:.2f}")

print(f"Average Turnaround Time = {avg\_tat:.2f}")

print("\nName")

# Run the function

sjf\_scheduling()



## **Conclusion: -**

Write your observation about Shortest Job algorithm. How it is better than First come first serve algorithm.

The algorithm schedules processes in the order in which the shortest job is done first. It has a minimum average waiting time. The average waiting time for given set of processes is minimum in SJF than FCFS which in turn leads to higher effectiveness of the system therefore its better than FCFS.