**Q1. Write a program to implement SJFscheduling algorithms for the following two cases:**

1. **When Arrival time is not same.**

**P# AT BT**

**P1 0 3**

**P2 2 6**

**P3 4 2**

**Implement thnote SJFscheduling(Non Preemptive) manually and verify your output.**

#include<stdio.h>

int main()

{

int i,n,p[10]={1,2,3,4,5,6,7,8,9,10},min,k=1,btime=0;

int bt[10],temp,j,at[10],wt[10],tt[10],ta=0,sum=0;

float wavg=0,tavg=0,tsum=0,wsum=0;

printf(" -------Shortest Job First Scheduling ( NP )-------\n");

printf("\nEnter the No. of processes :");

scanf("%d",&n);

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

{

printf("\tEnter the burst time of %d process :",i+1);

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

printf("\tEnter the arrival time of %d process :",i+1);

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

}

/\*Sorting According to Arrival Time\*/

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

{

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

{

if(at[i]<at[j])

{

temp=p[j];

p[j]=p[i];

p[i]=temp;

temp=at[j];

at[j]=at[i];

at[i]=temp;

temp=bt[j];

bt[j]=bt[i];

bt[i]=temp;

}

}

}

/\*Arranging the table according to Burst time,

Execution time and Arrival Time

Arrival time <= Execution time

\*/

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

{

btime=btime+bt[j];

min=bt[k];

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

{

if (btime>=at[i] && bt[i]<min)

{

temp=p[k];

p[k]=p[i];

p[i]=temp;

temp=at[k];

at[k]=at[i];

at[i]=temp;

temp=bt[k];

bt[k]=bt[i];

bt[i]=temp;

}

}

k++;

}

wt[0]=0;

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

{

sum=sum+bt[i-1];

wt[i]=sum-at[i];

wsum=wsum+wt[i];

}

wavg=(wsum/n);

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

{

ta=ta+bt[i];

tt[i]=ta-at[i];

tsum=tsum+tt[i];

}

tavg=(tsum/n);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\n RESULT:-");

printf("\nProcess\t Burst\t Arrival\t Waiting\t Turn-around" );

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

{

printf("\n p%d\t %d\t %d\t\t %d\t\t\t%d",p[i],bt[i],at[i],wt[i],tt[i]);

}

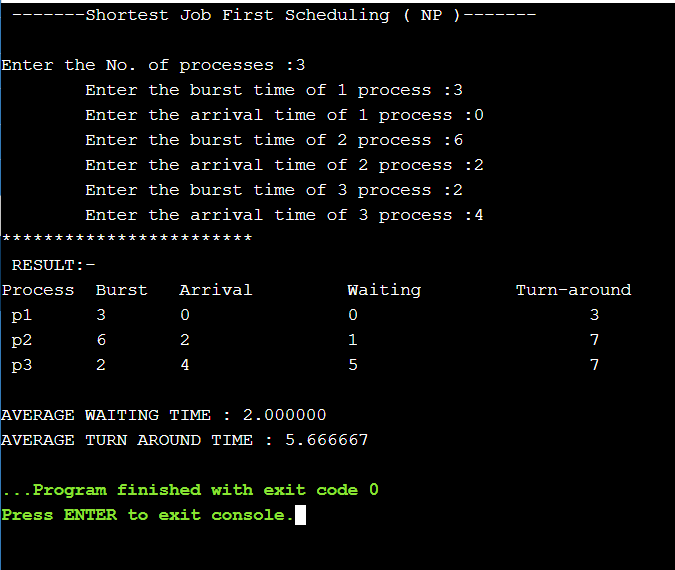
printf("\n\nAVERAGE WAITING TIME : %f",wavg);

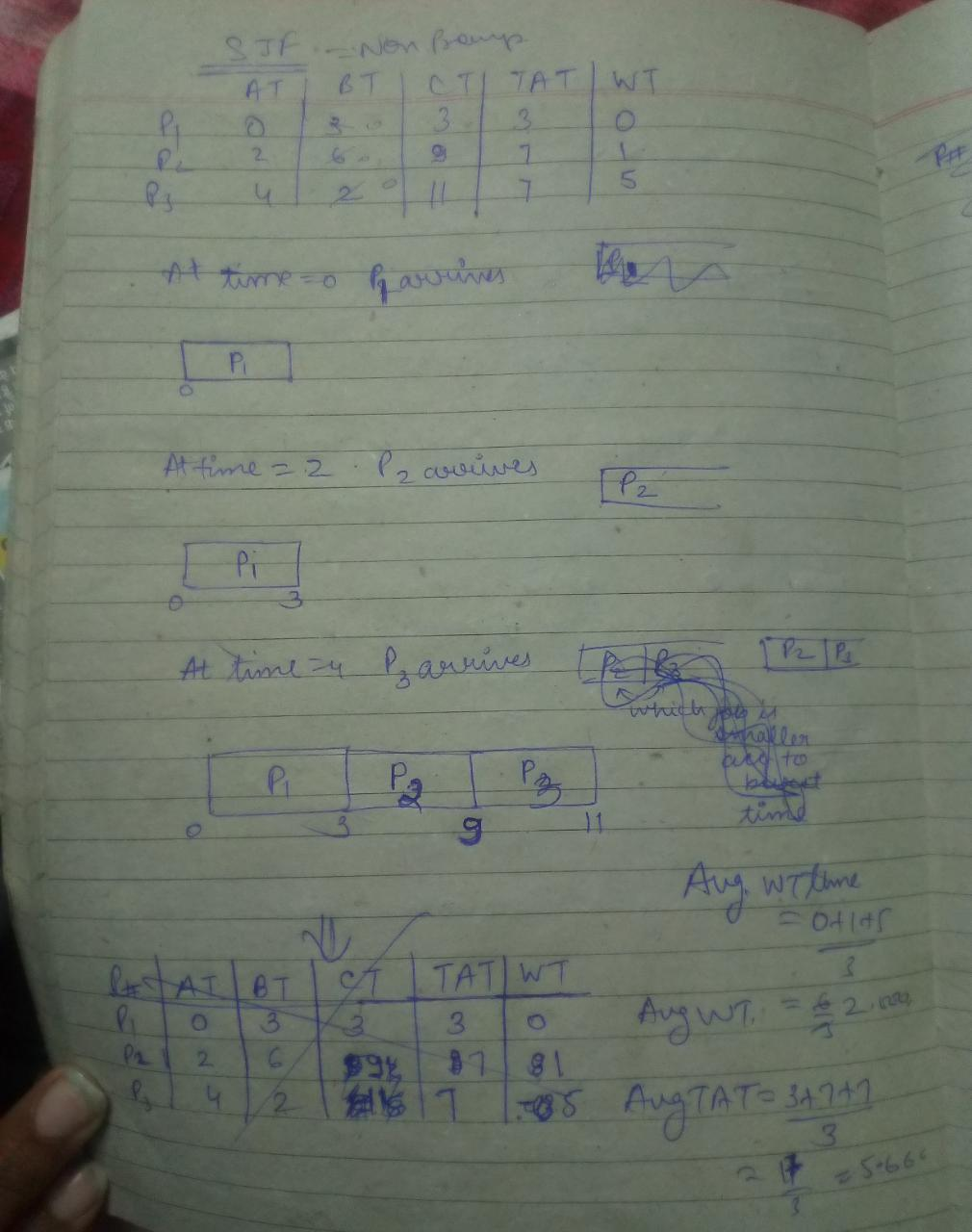
printf("\nAVERAGE TURN AROUND TIME : %f",tavg);

return 0;

}

**OUTPUT-**

****

****