Lab 7

I) WAP to illustrate priority scheduling in operation. The program accepts the total number of priorities, along with each packet priority, arrival time and burst time, and calculates the packet waiting time and turnaround time. Display all the timings and priority for each packet. Assume a non-preemptive priority queuing.

Program:

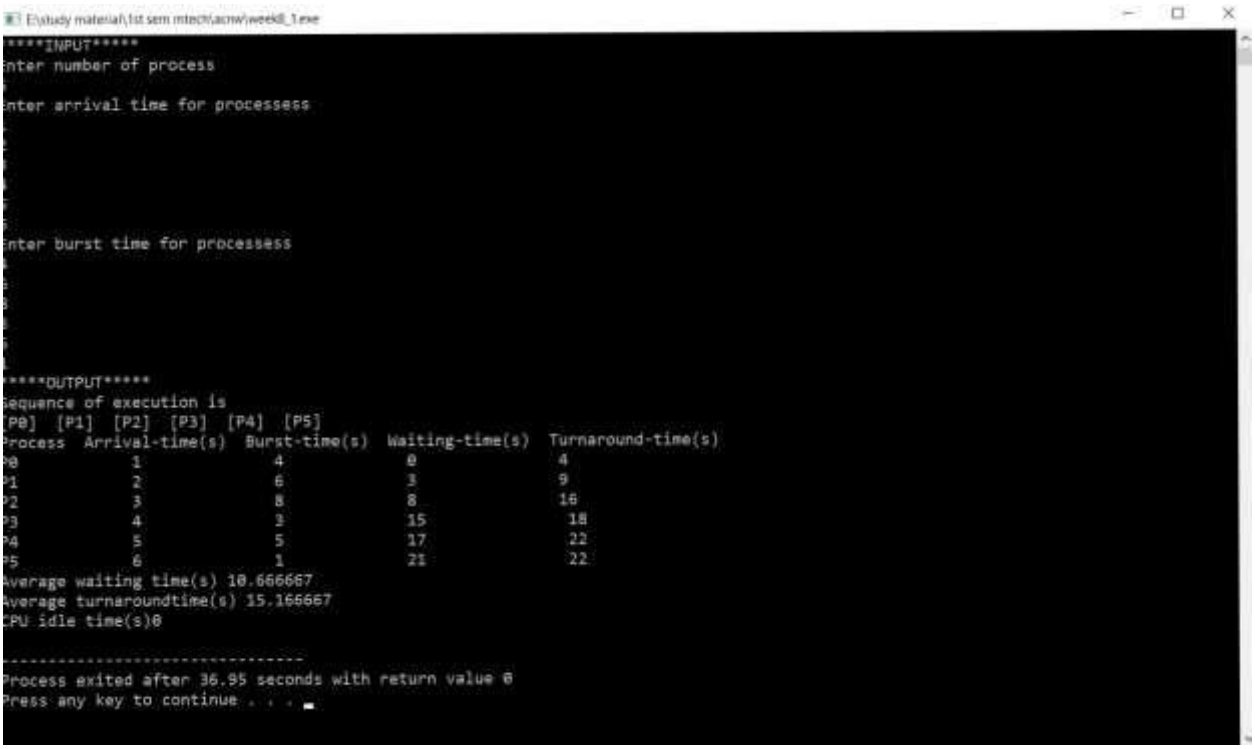
```
#include<stdio.h> int
main()
{
   Int at[10],at2[10],bt[100],ex[100],seq[100],re[100],wt[100],tat[100];
int n,i,j,start,pos,max=0,min,idle=0,k=0;     float av1=0,av2=0;

   printf("*****INPUT*****\n");
printf("Enter number of process\n");
scanf("%d",&n);
   printf("Enter arrival time for processess\n");
for(i=0;i<n;i++)
   {
    scanf("%d",&at[i]);
at2[i]=at[i];
   }
   printf("Enter burst time for processess\n");
for(i=0;i<n;i++)
   {
    scanf("%d",&bt[i]);
   }
   start=at[0];
for(i=1;i<n;i++)
   {
     if(start>at[i])
      {
      start=at[i];
      }
```

```
    }
   printf("*****OUTPUT*****\n");
printf("Sequence of execution is\n");
   for(i=0;i<n;i++)
   {
   if(max<at[i])
    {
     max=at[i];
    }
   }
   max=max+1;
for(i=0;i<n;i++,k++)
{  min=max;
for(j=0;j<n;j++){
if(at[j]!=-1)
          {
            if(at[j]<min)
              {
               min=at[j];
pos=j;
              }
            }
       }
     printf("[P%d]  ",pos);
seq[k]=pos;
if(start<at[pos]){
re[pos]=start;
idle+=at[pos]-start;
start=at[pos];
start+=bt[pos];
at[pos]=-1;
ex[pos]=start;
    }      else{
re[pos]=start;
```

```
start+=bt[pos];
at[pos]=-1;
       ex[pos]=start;
      }
    }
   printf("\n");
for(i=0;i<n;i++)
   {
     tat[i]=ex[i]-at2[i];
wt[i]=tat[i]-bt[i];
   }
 printf("Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)
Turnaroundtime(s)\n");    for(i=0;i<n;i++)
   {
    printf("P%d          %d           %d          %d
%d\n",i,at2[i],bt[i],wt[i],tat[i]);
   }
  for(i=0;i<n;i++)
  {
   av1+=tat[i];
av2+=wt[i];
  }
  printf("Average waiting time(s) %f\nAverage turnaroundtime(s) %f\nCPU
idle time(s)%d\n",av2/n,av1/n,idle);
}
```

Output:



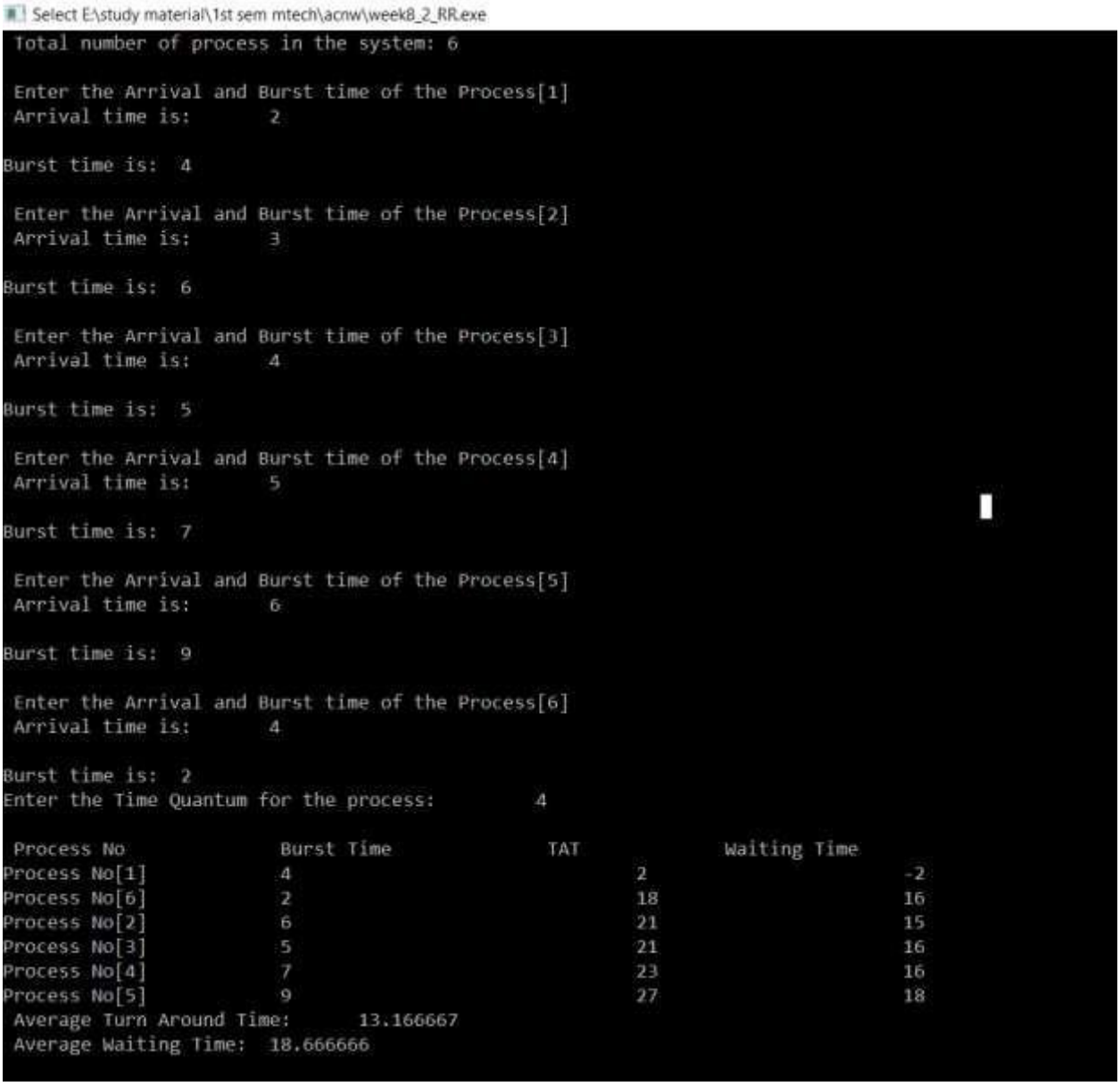
II) WAP to illustrateround robin scheduling in operation. The program accepts the total number of classes, along with each packet class, arrival time and burst time. The program calculates the packet departure time, delay between arrival and departure time, and the averagedelay for all packets and displaysall thesetimings along with the classfor each packet.Assume a work conserving policy.

Program:

```
#include<stdio.h>
#include<conio.h>
int main()
{
   // initlialize the variable name       int i, NOP, sum=0,count=0, y, quant,
wt=0, tat=0, at[10], bt[10], temp[10];       float avg_wt, avg_tat;       printf("
Total number of process in the system: ");       scanf("%d", &NOP);       y =
NOP; // Assign the number of process to variable y
  // Use for loop to enter the details of the process like Arrival time and the Burst
Time   for(i=0; i<NOP; i++)
{
printf("\n Enter the Arrival and Burst time of the Process[%d]\n", i+1);
printf(" Arrival time is: \t");  // Accept arrival time   scanf("%d",
&at[i]);   printf(" \nBurst time is: \t"); // Accept the Burst time
scanf("%d", &bt[i]);   temp[i] = bt[i]; // store the burst time in temp
array
}
// Accept the Time qunat   printf("Enter the Time
Quantum for the process: \t");   scanf("%d",
&quant);
// Display the process No, burst time, Turn Around Time and the waiting time
printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time ");
for(sum=0, i = 0; y!=0; )
{
if(temp[i] <= quant && temp[i] > 0) // define the conditions
{
   sum = sum + temp[i];
temp[i] = 0;
count=1;
   }
   else if(temp[i] > 0)
   {
```

```
      temp[i] = temp[i] - quant;
sum = sum + quant;
   }
   if(temp[i]==0 && count==1)
   {
      y--; //decrement the process no.
      printf("\nProcess No[%d] \t\t %d\t\t\t\t %d\t\t\t %d", i+1, bt[i], sum-at[i],
sum-at[i]-bt[i]);          wt = wt+sum-at[i]-bt[i];          tat = tat+sum-at[i];
count =0;
   }
   if(i==NOP-1)
   {
i=0;
   }
   else if(at[i+1]<=sum)
   {
i++;
   }
else
   {
i=0;
   }
}
// represents the average waiting time and Turn Around time
avg_wt = wt * 1.0/NOP;   avg_tat = tat * 1.0/NOP;   printf("\n
Average Turn Around Time: \t%f", avg_wt);   printf("\n
Average Waiting Time: \t%f", avg_tat);   getch();
}
```

Output:


Select E:\study material\1st sem mtech\acnw\week8_2_RR.exe
Total number of process in the system: 6
Enter the Arrival and Burst time of the Process[1]
Arrival time is: 2
Burst time is: 4
Enter the Arrival and Burst time of the Process[2]
Arrival time is: 3
Burst time is: 6
Enter the Arrival and Burst time of the Process[3]
Arrival time is: 4
Burst time is: 5
Enter the Arrival and Burst time of the Process[4]
Arrival time is: 5
Burst time is: 7
Enter the Arrival and Burst time of the Process[5]
Arrival time is: 6
Burst time is: 9
Enter the Arrival and Burst time of the Process[6]
Arrival time is: 4
Burst time is: 2
Enter the Time Quantum for the process: 4
Process No Burst Time TAT Waiting Time
Process No[1] 4 2 -2
Process No[6] 2 18 16
Process No[2] 6 21 15
Process No[3] 5 21 16
Process No[4] 7 23 16
Process No[5] 9 27 18
Average Turn Around Time: 13.166667
Average Waiting Time: 18.666666