ACNW LAB 8

Q1. 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.

CODE:

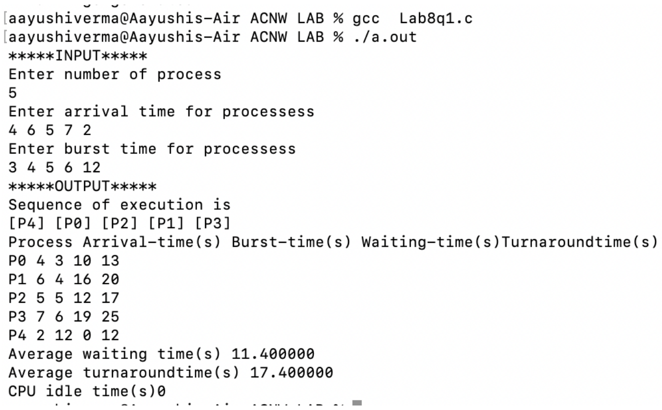
```
#include<stdio.h>



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:



```
aayushiverma@Aayushis-Air ACNW LAB % gcc  Lab8q1.c
aayushiverma@Aayushis-Air ACNW LAB % ./a.out
*****INPUT*****
Enter number of process
5
Enter arrival time for processess
4 6 5 7 2
Enter burst time for processess
3 4 5 6 12
*****OUTPUT*****
Sequence of execution is
[P4] [P0] [P2] [P1] [P3]
Process Arrival-time(s) Burst-time(s) Waiting-time(s)Turnaroundtime(s)
P0 4 3 10 13
P1 6 4 16 20
P2 5 5 12 17
P3 7 6 19 25
P4 2 12 0 12
Average waiting time(s) 11.400000
Average turnaroundtime(s) 17.400000
CPU idle time(s)0
```

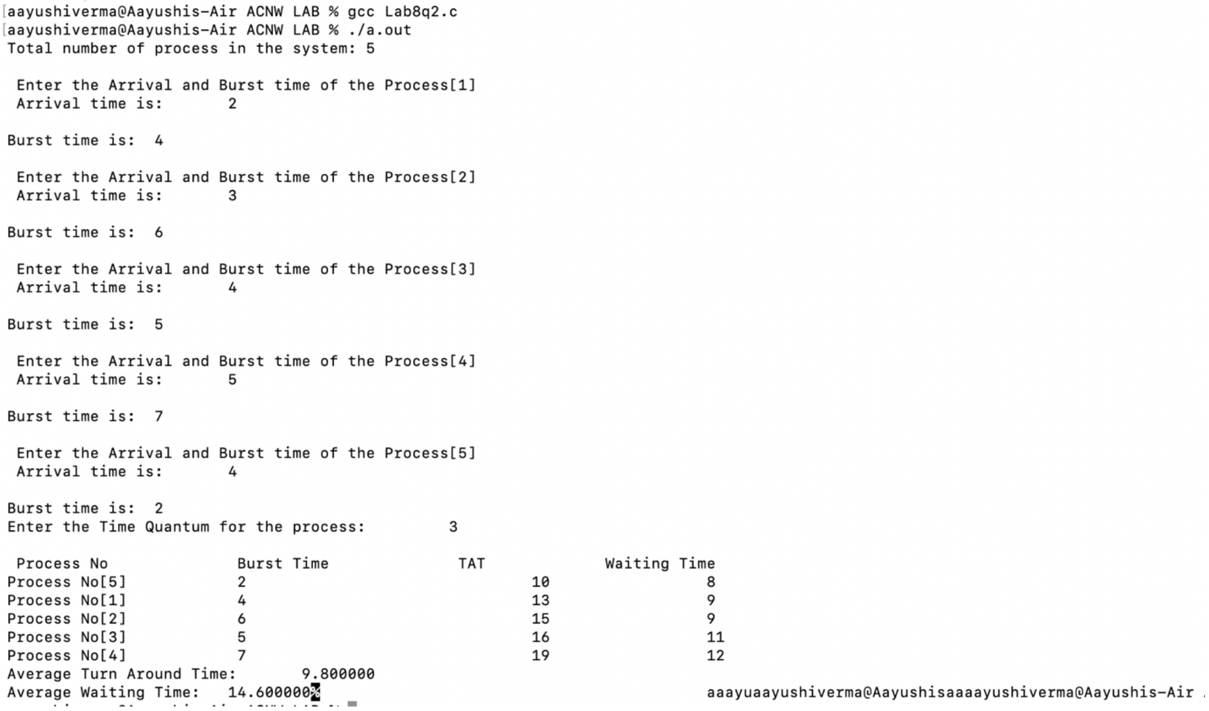
Q2 WAP to illustrate round robin scheduling in operation. The program accepts the total number of classes, along with each packet class, arrival time and burst time. The program calculates the packet departure time, delay between arrival and departure time, and the average delay for all packets and displays all these timings along with the class for each packet. Assume a work conserving policy.

CODE:

```
#include<stdio.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 BurstTime
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("\nAverage Turn Around Time: \t%f", avg_wt);
    printf("\nAverage Waiting Time: \t%f", avg_tat);


}
```

OUTPUT:



```
aayushiverma@Aayushis-Air ACNW LAB % gcc Lab8q2.c
aayushiverma@Aayushis-Air ACNW LAB % ./a.out
Total number of process in the system: 5

 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:       4

Burst time is:  2
Enter the Time Quantum for the process:         3

 Process No              Burst Time              TAT             Waiting Time
Process No[5]            2                               10                 8
Process No[1]            4                               13                 9
Process No[2]            6                               15                 9
Process No[3]            5                               16                 11
Process No[4]            7                               19                 12
Average Turn Around Time:       9.800000
Average Waiting Time:   14.600000%                                          aaayuaayushiverma@Aayushisaaaayushiverma@Aayushis-Air
```