LAB - 3

Write a C program to simulate the following CPU scheduling algorithm to find turnaround time and waiting time.
Priority (pre-emptive or Non-pre-emptive)
Round Robin (Experiment with different quantum sizes for RR algorithm)

CODE:
Priority (pre-emptive or Non-pre-emptive)

```
#include<stdio.h>
int at[10],t,pt[10],tat[10],wt[10],n,time=0,i,ready[10],pry[10],op=0, maxpr,x,p[10];
float atat=0,awt=0;

void main()
{
   printf("Enter number of processes \n");
   scanf("%d",&n);

   printf("Enter araival times: \n");
   for(i=0;i<n;i++)
   scanf("%d",&at[i]);

   printf("Enter process times: \n");
   for(i=0;i<n;i++)
   scanf("%d",&pt[i]);

   printf("Enter priority: \n");
   for(i=0;i<n;i++)
   scanf("%d",&pry[i]);

   for(i=0;i<n;i++)
   ready[i]=0;

   for(i=0;i<n;i++)
   p[i]=pt[i];

   for(i=0;i<n;i++)
   time+=pt[i];
   t=n;
   while(t--)
   {
```

```
    for(i=0;i<n;i++)
    if(op>=at[i])
    ready[i]=1;

    for(i=0;i<n;i++)
    if(pt[i]==0)
    pry[i]=0;


    //finding index of max priority
    maxpr=pry[0];
    for(i=0;i<n;i++)
    if(ready[i]==1)
    if(pry[i]>maxpr)
    maxpr=pry[i];

    for(i=0;i<n;i++)
    if(maxpr==pry[i])
    x=i;

    //printing chart
    printf("%d p%d ",op,(x+1));


    op=op+pt[x];
    tat[x]=op;
    ready[x]=0;
    pry[x]=0;
}
printf("%d",op);

//finding avgtat and avg wt
for(i=0;i<n;i++)
{
    tat[i]=tat[i]-at[i];
}

for(i=0;i<n;i++)
{
    atat+=tat[i];
    wt[i]=tat[i]-pt[i];
}
for(i=0;i<n;i++)
awt+=wt[i];
```

```
    awt=awt/n;
    atat=atat/n;

    //printing final values
    printf("\n");
    for(i=0;i<n;i++)
    printf("P%d  %d  %d \n",(i+1),tat[i],wt[i]);
    printf("ATAT=%f  \nAWT=%f ",atat,awt);
}
```

Round Robin

```
#include<stdio.h>

    int tq, at[10], pt[10], p[10], time=0, op=0, i,j ,n, ready[10],q[100];
    int r=-1,f=0,tat[10],wt[10],z,fg,y=9999,ch;
    float atat,awt;

int rr(int x)
{
    if(pt[x]>tq)
    {
        pt[x]-=tq;
        op+=tq;
    }
    else
    {
        op+=pt[x];
        pt[x]=0;
        tat[x]=op;
        ready[x]=0;
    }
    return x;
}

void main()
{
    printf("Enter number or processes \n");
    scanf("%d",&n);

    printf("Enter araival times: \n");
    for(i=0;i<n;i++)
    scanf("%d",&at[i]);
```

```
printf("Enter process times: \n");
for(i=0;i<n;i++)
scanf("%d",&pt[i]);

printf("Enter TQ \n");
scanf("%d",&tq);

for(i=0;i<n;i++)
ready[i]=0;

for(i=0;i<n;i++)
q[i]=9999;

for(i=0;i<n;i++)
p[i]=pt[i];

for(i=0;i<n;i++)
time+=pt[i];


for(i=0;i<n;i++)
    if(op>=at[i])
    ready[i]=1;

for(i=0;i<n;i++)
    if(ready[i]==1)
    {
        q[++r]=i;
    }

while(op!=time)
{
    printf("%d ",op);
    if(z==y)
    q[++f];
    y=z;

    ch=q[f];
    if(pt[ch]!=0)
    {
    z=rr(q[f]);

    printf("P%d ",(z+1));
```

```
        for(i=0;i<n;i++)
        {
           if(op>=at[i] && pt[i]!=0)
           {
            fg=0;
            j=f;
            while(j<=r)
            {
                if(i==q[j])
                fg=1;
                j++;
            }
            if(fg==0)
            {
                q[++r]=i;
            }
           }
        }
        if(pt[z]!=0)
        q[++r]=z;
        }
        f++;
    }

    printf("%d ",op);

    for(i=0;i<n;i++)
    {
        tat[i]=tat[i]-at[i];
        wt[i]=tat[i]-p[i];
        atat+=tat[i];
        awt+=wt[i];
    }
    atat=atat/n;
    awt=awt/n;

    printf("\n");
    for(i=0;i<n;i++)
    printf("P%d %d %d \n",(i+1),tat[i],wt[i]);
    printf("ATAT=%f \nAWT=%f ",atat,awt);
}
```

OUTPUT:

PRIORITY OUTPUT:

```
PS D:\VS Code\OS> cd "d:\VS Code\OS\" ; if ($?) { gcc npp.c -o npp } ; if ($?) { .\npp }
Enter number of processes
4
Enter araival times:
0 1 2 3
Enter process times:
4 3 3 5
Enter priority:
3 4 6 5
0 p1 4 p3 7 p4 12 p2 15
P1  4  0
P2  14  11
P3  5  2
P4  9  4
ATAT=8.000000
AWT=4.250000
PS D:\VS Code\OS>
```

ROUND ROBIN OUTPUT:

```
C:\Users\Admin\Desktop\1BM21CS010\1BM22CS404\bin\Debug\1BM22CS404.exe
Enter Total Process:     5
Enter Arrival Time and Burst Time for Process Process Number 1 :0 5
Enter Arrival Time and Burst Time for Process Process Number 2 :1 3
Enter Arrival Time and Burst Time for Process Process Number 3 :2 1
Enter Arrival Time and Burst Time for Process Process Number 4 :3 2
Enter Arrival Time and Burst Time for Process Process Number 5 :4 3
Enter Time Quantum:     2

Process |Turnaround Time|Waiting Time

P[3]    |       3       |       2
P[4]    |       4       |       2
P[2]    |       11      |       8
P[5]    |       9       |       6
P[1]    |       14      |       9

Average Waiting Time= 5.400000
Avg Turnaround Time = 8.200000
Process returned 0 (0x0)   execution time : 74.742 s
Press any key to continue.
```