Assignment- 10
Name: Tanmay Pol Roll no. 33 Div: TY CS C

Problem Statement: Write a program to implement Round Robin Algorithm.

Code:

```
#include<bits/stdc++.h>

usingnamespacestd;


/* C++ Program to Round Robin*/


typedefstructprocess

{

    int id, at,bt, st,ct, pr;

    floatwt, tat;

}process;


process  [10],     [10],          ;

queue<int>     ;


intaccept(intch);

voidturnwait(intn);

voiddisplay(intn);

voidganttrr(intn);
```

```
intmain()
{
    inti,n,ts,ch,j,x;

    p[0].tat =0;
    p[0].wt=0;

    n=accept(ch);
    ganttrr(n);
    turnwait(n);

    display(n);

    return0;
}

intaccept(intch)
{
    int ,  ;

    printf("Enter the Total Number of Process: ");
    scanf("%d",&  );
```

```
if(n==0)
{
    printf("Invalid");
    exit(1);
}

cout<<endl;

for(i=1;i<=n;i++)
{
    printf("Enter arrival time of the Process P%d: ",i);
    scanf("%d",&p[i].at);
    p[i].id =i;
}
```

```
        <<endl;

for( =1; <=  ; ++)
{
    printf("Enter burst time of the Process P%d: ", );
    scanf("%d" &  [ ].bt);
}

for( =1; <=  ; ++)
```

```
    {
        p1[i]=p[i];
    }
    returnn;
}

voidganttrr(intn)
{
    inti,ts,m,nextval,nextarr;

    nextval=p1[1].at;
    i=1;

    cout<<"\nEnter the Time Quantum: ";
        >>  ;

    for( =1, <=n&&    [ ].at <=          ; ++)
    {
          .push(    [ ].id);
    }

    while(!    empty())
    {
        =    .front();
```

```
q1.pop();


if(p1[m].bt>=ts)
{
    nextval=nextval+ts;
}
else
{
    nextval=nextval+p1[m].bt;
}
if(p1[m].bt>=ts)
{
    p1[m].bt=p1[m].bt-ts;
}
else
{
        [   ].bt=0;
}

while( <=n&&    [ ].at <=           )
{
        .push(    [ ].id);
    ++;
}
```

```
        if(p1[m].bt>0)
        {
            q1.push(m);
        }
        if(p1[m].bt<=0)
        {
            p[m].ct=nextval;
        }
    }
}

voidturnwait(intn)
{
    int ;

    for( =1; <=n; ++)
    {
        [ ].tat=  [ ].ct-  [ ].at;
        [ ].wt=  [ ].tat -  [ ].bt;
        [0].tat =  [0].tat +  [ ].tat;
        [0].wt=  [0].wt+  [ ].wt;
    }
```

```
    p[0].tat =p[0].tat /n;
    p[0].wt=p[0].wt/n;
}

voiddisplay(intn)
{
    inti;
    printf("\nProcess\tAT\tBT\tCT\tTAT\t\tWT");

    for(i=1;i<=n;i++)
    {
        printf("\nP%d\t%d\t%d\t%d\t%f\t%f",  [ ].id,  [ ].at,  [ ].bt,  [ ].ct,  [ ].tat,  [ ].wt);
    }
        <<"\n=================================================\n";
    printf("\nAverage Turn Around Time: %f",  [0].tat);
    printf("\nAverage Waiting Time: %f\n",  [0].wt);
}
```

Output:

```
Enter the Total Number of Process: 4

Enter arrival time of the Process P1: 0
Enter arrival time of the Process P2: 1
Enter arrival time of the Process P3: 2
Enter arrival time of the Process P4: 4

Enter burst time of the Process P1: 5
Enter burst time of the Process P2: 4
Enter burst time of the Process P3: 2
Enter burst time of the Process P4: 1

Enter the Time Quantum: 2

Process AT      BT      CT      TAT             WT
P1      0       5       12      12.000000       7.000000
P2      1       4       11      10.000000       6.000000
P3      2       2       6       4.000000        2.000000
P4      4       1       9       5.000000        4.000000
================================================

Average Turn Around Time: 7.750000
Average Waiting Time: 4.750000
```