Design, develop and implement program to simulate the working of Round Robin (RR) scheduling algorithms. Experiment with different quantum sizes for RR algorithm.

Algorithm

1. Start

2. Define a structure J to represent a job with attributes: bt (burst time), tat (turnaround time), wt (waiting time), at (arrival time), ft (finish time).

3. Define an array of job structures named 'job' to store job details.

4. Define a function scheduler(job[], n, q) where:
   a. Initialize an array 'burst' to store burst times.
   b. Initialize variables: t = 0, done = 0, curr = -1, diff = q, i = 0.
   c. Copy burst times from jobs to the 'burst' array.
   d. While not all jobs are done:
      i. Iterate through jobs in a circular manner to find the next non-empty job:
         a. Increment 'curr' in a circular manner.
         b. If the burst time of the current job is not zero, break from the loop.

      ii. Calculate the time to be executed for the current job:
         diff = min(q, job[curr].bt).

      iii. Update the burst time of the current job and the current time:
         job[curr].bt -= diff.
         t += diff.

      iv. If the burst time of the current job becomes zero:
         a. Increment 'done'.
         b. Set the finish time of the current job to the current time.

   e. Print "RR Scheduling Details are".

   f. Calculate turnaround time, waiting time, and reset burst times for each job:
      i. Iterate through jobs:
         a. Reset burst time to the original value.
         b. Calculate turnaround time: tat = finish time - arrival time.
         c. Calculate waiting time: wt = turnaround time - burst time.
         d. Accumulate turnaround time and waiting time for later calculation.

   g. Print the details of each job (Job, BT, AT, TAT, WT).

   h. Print average turnaround time and average waiting time:
      i. avg_tat = sum of turnaround times / number of jobs.

ii. avg_wt = sum of waiting times / number of jobs.
iii. Print "Avg TAT = avg_tat" and "Avg WT = avg_wt".

5. Define the main function:
a. Initialize variables: n, q, c, i.
b. Print "Enter the number of processes:" and read input for 'n'.

c. Print "Enter the arrival time and burst time":
i. Iterate 'i' from 0 to 'n - 1':
a. Print "Job i + 1:" and read 'job[i].at' and 'job[i].bt'.

d. Print "Enter time quantum:" and read input for 'q'.

e. Call the scheduler function: scheduler(job, n, q).

6. End

//PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
struct J
{
  int bt,tat,wt,at,ft;
} job[100];
void scheduler(struct J job[],int n,int q)
{
        int burst[100],t=0,done=0,curr=-1,diff=q,i=0;
        float tat_sum=0,wt_sum=0;
        for(i=0;i<n;i++)
         burst[i]=job[i].bt;

        while(done<n)
        {
                 while(1)
                {
                        curr=(curr+1)%n;
                        if(job[curr].bt!=0)
                                break;
                }
                diff=(q<=job[curr].bt)?q:job[curr].bt;
```

```
                job[curr].bt-=diff;
                t+=diff;
                if(job[curr].bt==0)
                {
                        done++;
                        job[curr].ft=t;
                }
        }
         printf("RR Scheduling Details are \n");
        for(i=0;i<n;i++)
        {
          job[i].bt=burst[i];
          job[i].tat=job[i].ft-job[i].at;
          job[i].wt=job[i].tat-job[i].bt;
          tat_sum+=job[i].tat;
          wt_sum+=job[i].wt;
        }
        printf("Job\tBT\tAT\tTAT\tWT\n");
        for(i=0;i<n;i++)
         printf("%d\t%d\t%d\t%d\t%d\n",i+1,job[i].bt,job[i].at,job[i].tat,job[i].wt);
        printf("Avg TAT=%f\nAvg WT=%f\n",tat_sum/n,wt_sum/n);
}
void main()
{
        int n,q,c,i;
        printf("Enter the number of processes:\n");
        scanf("%d",&n);
        printf("Enter the arrival time and burst time\n");
        for(i=0;i<n;i++)
        {
                printf("Job%d: ",i+1);
                scanf("%d%d",&job[i].at,&job[i].bt);
        }

        printf("Enter time quantum: ");
        scanf("%d",&q);
        scheduler(job,n,q);
}
```

OUTPUT

```
Enter the number of processes:
4
```

Enter the arrival time and burst time
Job1: 0 8
Job2: 1 4
Job3: 2 9
Job4: 3 5
Enter time quantum: 3
RR Scheduling Details are

Job	BT	AT	TAT	WT
1	8	0	23	15
2	4	1	15	11
3	9	2	24	15
4	5	3	18	13

Avg TAT=20.000000
Avg WT=13.500000