Aim :- Implementation of Round Robin schedulbing algorithm

Round Robin schedulbing algorithm

Round Robin schedulling algorithm is one of the most popular scheduling algorithm which can actully be implemented in most of the operating system. this is the preemptive version of first come first serve scheduling. The algorithm focuses on time sharing. In this algorithm every process gets executed in a cyclic way. A certain time slice is defined in the system which is called time quantum. Each process present in the ready queue. is assigned the cpu for that time quantum, if the process will terminate else the process will go back to the ready queue and waits for the next turn to complete the execution.

Advantages

- It can be actually implementable in the system because it is not depending on the burst time
- It doesn't suffer from the problem of starvation or convey effect
- All the jobs get a fare allocation of cpu

Disadvantages

- The higher time quantum, the higher the response time in the system
- The lower the time quantum, the higher the context switching overhead in the system.
- Deciding a perfect time quantum is really a very difficult

task in the system.

Algorithm :

process	Burst time
P_1	2
P_2	1
P_3	8
P_4	4
P_5	3

(quantum = 2)

quantum is : then table look like

process	Burst time.
P_1	2
P_2	1
P_3	2
P_4	2
P_5	2
P_3	2
P_4	2
P_5	1
P_3	2
P_3	2

2	1	2	2	2	2	2	1	2	2	
P_1	P_2	P_3	P_4	P_5	P_3	P_4	P_5	P_3	P_3	
0	2	3	5	7	9	11	13	14	16	18

$P_1 = 0$

$P2 = 2$

$P_3 =$

$P_4 =$

$P_5 =$

Conclusion :- Thus we study Implementation of Round Robin Scheduling algorithm

13/11/19

Program:

```
#include<iostream>
using namespace std;
#include<conio.h>
struct process
{
	int no;
	int at,et,wt,tt;
	int tet;
	int t;
};

int main()
{
	process p[99];
	int i,j,k;
	cout<<"\n Enter No of Processes:";
	int np;
	cin>>np;

	for (i=0;i<np;i++)
	{
		cout<<"\n Enter Execution time of process"<<i+1<<":";
		cin>>p[i].et;
		p[i].tet=p[i].et;
		p[i].at=p[i].t=p[i].tt=p[i].wt=0;
		p[i].no=i+1;
	}

	cout<<"\n Enter Time Quantum:";
	int q;
	cin>>q;

	cout<<"\n Entered Data";
	cout<<"\n Process\tET";
	for(i=0;i<np;i++)
	{
		cout<<"\n "<<p[i].no<<"\t"<<p[i].et;
	}

	int totaltime=0;
	for(i=0;i<np;i++)
	{
		totaltime+=p[i].et;
	}

	i=0;
	k=0;

	int rrg[99];
	for(j=0;j<totaltime;j++)
	{
		if((k==0)&&(p[i].et!=0))
```

```
        {       p[i].wt=j;
                if((p[i].t!=0))
                {
                        p[i].wt-=q*p[i].t;
                }
        }
        if((p[i].et!=0)&&(k!=q))
        {
                rrg[j]=p[i].no;
                p[i].et-=1;
                k++;
        }
        else
        {
                if((k==q)&&(p[i].et!=0))
                {
                        p[i].t+=1;
                }
                i=i+1;
                if(i==np)
                {
                        i=0;
                }

                        k=0;
                j=j-1;
        }
    }
    int twt=0;
    int ttt=0;
    cout<<"\n Result Of Round Robin";
    cout<<"\n PNo\tET\tWT\tTT";
    for(i=0;i<np;i++)
    {
        p[i].tt=p[i].wt+p[i].tet;
        ttt+=p[i].tt;
        twt+=p[i].wt;
        cout<<"\n "<<p[i].no<<"\t"<<"\t"<<p[i].tet<<"\t"<<p[i].wt<<"\t"<<p[i].tt;
    }
    cout<<"\n Average Waiting Time:"<<(float)twt/np;
    cout<<"\n Average Turn Around Time:"<<(float)ttt/np;

    getch();
}
```

Output:

```
 Enter No of Processes:3

 Enter Execution time of process1:12

 Enter Execution time of process2:55
```

```
Enter Execution time of process3:26

Enter Time Quantum:10

Entered Data
Process         ET
1       12
2       55
3       26
Result Of Round Robin
PNo     ET          WT          TT
1                   12          20          32
2                   55          38          93
3                   26          42          68
Average Waiting Time:33.3333
Average Turn Around Time:64.3333
```

✓

13/11/19