**CPU SCHEDULING-ROUND ROBIN**

SANJANA S NAIR

CSE C

33

#include<stdio.h>

#include<string.h>

int q[50],f=-1,r=-1;

struct process

{

int at,bt,wt,tt,stat,rt;

char name[10];

} p[10];

struct gantt

{

int st,ct;

char name[10];

} g[10];

void enqueue(int a)

{

if(f==-1)

f=0;

q[++r]=a;

}

int dequeue()

{

if(f==-1 && r==-1)

return -1;

else if(f==r)

{

int a=q[f];

f=-1;

r=-1;

return a;

}

else

{

return q[f++];

}

}

void main()

{

int n,i,j=0,idle=0,ls=0,m=0,t,num=0,k;

float avgtt=0,avgwt=0;

printf("Enter the number of process : ");

scanf("%d",&n);

for(i=0; i<n; i++)

{

printf("Enter the process name : ");

scanf("%s",p[i].name);

printf("Enter the arrival time : ");

scanf("%d",&p[i].at);

printf("Enter the burst time : ");

scanf("%d",&p[i].bt);

p[i].stat=0;

p[i].rt=p[i].bt;

}

printf("Enter the time quantum : ");

scanf("%d",&t);

i=0;

while(ls<n)

{

for(j=0; j<n; j++)

{

if((p[j].stat==0)&&(p[j].at<=i))

{

enqueue(j);

p[j].stat=1;

}

}

if((idle==0)&&(f==-1))

{

strcpy(g[num].name,"idle");

g[num].st=i;

idle=1;

I++;

}

else if(f!=-1)

{

if(idle==1)

{

g[num].ct=i;

idle=0;

Num++;

}

k=dequeue();

g[num].st=i;

strcpy(g[num].name,p[k].name);

if(p[k].rt<=t)

{

g[num].ct=i+p[k].rt;

i=g[num].ct;

p[k].tt=i-p[k].at;

p[k].wt=p[k].tt-p[k].bt;

p[k].stat=2;

ls++;

num++;

}

else

{

g[num].ct=i+t;

i=i+t;

p[k].rt=p[k].rt-t;

num++;

for(m=0; m<n; m++)

{

if((p[m].stat==0)&&(p[m].at<=i))

{

enqueue(m);

p[m].stat=1;

}

}

enqueue(k);

}

}

else

{

i++;

}

}

printf("\nPR\tAT \tBT \tWT \tTT\n");

for(i=0; i<n; i++)

{

printf("%s\t %d \t%d\t%d\t%d\n",p[i].name,p[i].at,p[i].bt,p[i].wt,p[i].tt);

avgwt+=p[i].wt;

avgtt+=p[i].tt;

}

printf("\nGantt chart");

printf("|");

for(m=0; m<num; m++)

{

if(strcmp(g[m].name,g[m+1].name)==0)

{

continue;

}

else

{

printf("%s\t|",g[m].name);

}

}

printf("%d\t",g[0].st);

for(m=0; m<num; m++)

{

if(strcmp(g[m].name,g[m+1].name)==0)

{

continue;

}

else

{

printf("%d\t",g[m].ct);

}

}

printf("\nAverage waiting time = %0.2f",avgwt/n);

printf("\nAverage turnaround time = %0.2f\n",avgtt/n);

}

**SAMPLE OUTPUT**

Enter the number of process : 4

Enter the process name : a

Enter the arrival time : 0

Enter the burst time : 8

Enter the process name : b

Enter the arrival time : 1

Enter the burst time : 4

Enter the process name : c

Enter the arrival time : 2

Enter the burst time : 1

Enter the process name : d

Enter the arrival time : 3

Enter the burst time : 5

Enter the time slice : 2

Process Table

| Name | Arrival time | Burst time | Waiting time | Turnaround

| a | 0 | 8 | 9 | 17

| b | 1 | 4 | 6 | 10

| c | 2 | 1 | 2 | 3

| d | 3 | 5 | 10 | 15

Gantt Chart

| a | b | c | a | d | b | a | d | a | d |

0 2 4 5 7 9 11 13 15 17 18

Average Turnaround Time : 11.250000

Average Waiting Time : 6.750000