**CPU SCHEDULING-SJF**

SANJANA S NAIR

CSE C

33

#include<stdio.h>

#include<string.h>

struct process

{

int at,bt,ct,tt,wt;

char name[20];

}p[20],temp;

void main()

{

int n,i,j,k=0,time=0,gtime=0;

printf("enter number of proceses\n");

scanf("%d",&n);

printf("enter processes\n");

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

{

printf("enter process name \n");

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

printf("enter arrival time\n");

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

printf("enter burst time\n");

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

}

printf("process table\n");

printf("pid\tat\tbt\n");

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

{

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

}

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

{

for(j=i+1;j<n;j++)

{

if(p[i].at>p[j].at)

{

temp=p[i];

p[i]=p[j];

p[j]=temp;

}

}

}

i=0,j=0;

while(i<n)

{

if(time>=p[i].at)

{

j=i+1;

while(j<n)

{

if(time>=p[i].at && p[i].bt>p[j].bt)

{

temp=p[i];

p[i]=p[j];

p[j]=temp;

}

j++;

}

time=time+p[i].bt;

p[i].ct=time;

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

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

i++;

}

else

{

time=p[i].at;

}

}

printf("updated process table\n");

printf("pid\tat\tbt\tct\ttt\twt\n");

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

{

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

}

while(k<n)

{

if(gtime>=p[k].at)

{

printf("| %s\t",p[k].name);

gtime=gtime+p[k].bt;

k++;

}

else

{

printf("| idle\t");

gtime = p[k].at;

}

}

printf("|\n|0\t|");

k=0,gtime=0;

while(k<n)

{

if(gtime>=p[k].at)

{

printf("%d\t",p[k].ct);

gtime=gtime+p[k].bt;

k++;

}

else

{

gtime=p[k].at;

printf("%d\t",gtime);

}

}

**}**

**SAMPLE OUTPUT**

Enter the no of process: 3

Enter the process name: p1

Enter the arrival time: 0

Enter the burst time: 2

Enter the process name: p2

Enter the arrival time: 2

Enter the burst time: 2

Enter the process name: p3

Enter the arrival time: 2

Enter the burst time: 1

PR AT BT CT TAT WT

p1 0 2 2 2 0

p2 2 2 5 3 1

p3 2 1 3 1 0

Gantt chart

| p1 | p3 | p2 |

0 2 3 5

Average turnaround time=2.00

Average waiting time=0.33