LAB 02

c)ROUND ROBIN:

To simulate the CPU scheduling algorithm round-robin.

#include<stdio.h>

void main()

{

Int i,j,n,bu[10],wa[10],tat[10],t,ct[10],max;

float awt=0,att=0,temp=0;

printf("Enter the no of processes -- ");

scanf("%d",&n);

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

{

printf("\nEnter Burst Time for process %d -- ", i+1);

scanf("%d",&bu[i]);

ct[i]=bu[i];

}

printf("\nEnter the size of time slice -- ");

scanf("%d",&t);

max=bu[0];

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

if(max<bu[i])

max=bu[i];

for(j=0;j<(max/t)+1;j++){

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

if(bu[i]!=0){

if(bu[i]<=t)

{ tat[i]=temp+bu[i];

temp=temp+bu[i];

bu[i]=0;

}

else

{ bu[i]=bu[i]-t;

temp=temp+t;

}}}}

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

{ wa[i]=tat[i]-ct[i];

att+=tat[i];

awt+=wa[i];}

printf("\nThe Average Turnaround time is -- %f",att/n);

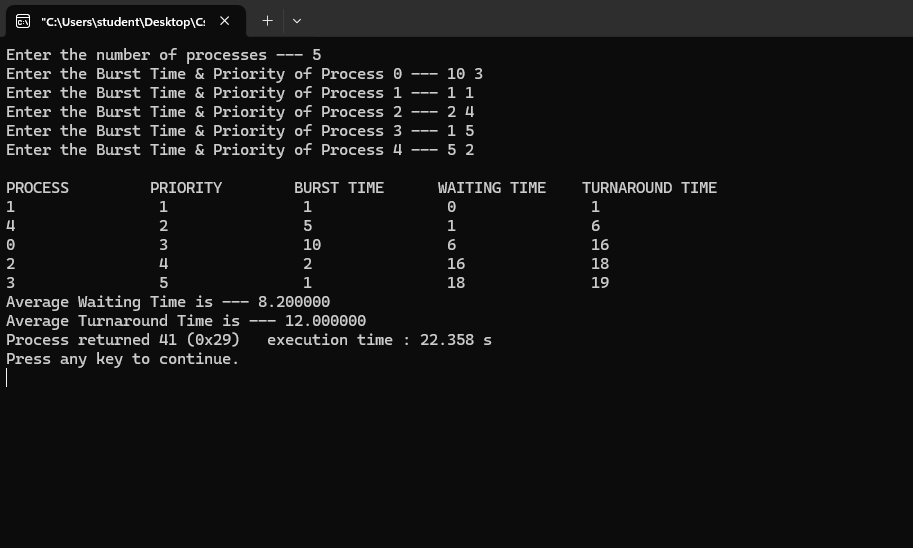
printf("\nThe Average Waiting time is -- %f ",awt/n);

printf("\n\tPROCESS\t BURST TIME \t WAITING TIME\tTURNAROUND TIME\n");

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

printf("\t%d \t %d \t\t %d \t\t %d \n",i+1,ct[i],wa[i],tat[i]);

OUTPUT:



d)PRIORITY:

To write a c program to simulate the CPU scheduling priority algorithm.

#include<stdio.h>

void main()

{

int p[20],bt[20],pri[20], wt[20],tat[20],i, k, n, temp; float wtavg, tatavg;

printf("Enter the number of processes --- "); scanf("%d",&n);

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

p[i] = i;

printf("Enter the Burst Time & Priority of Process %d --- ",i);

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

}

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

for(k=i+1;k<n;k++) {

if(pri[i] > pri[k])

{ temp=p[i]; p[i]=p[k];

p[k]=temp; temp=bt[i];

bt[i]=bt[k];

bt[k]=temp;

temp=pri[i];

pri[i]=pri[k];

pri[k]=temp;

}}}

wtavg = wt[0] = 0;

tatavg = tat[0] = bt[0];

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

{

wt[i] = wt[i-1] + bt[i-1];

tat[i] = tat[i-1] + bt[i];

wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

printf("\nPROCESS\t\tPRIORITY\tBURST TIME\tWAITING TIME\tTURNAROUND TIME");

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

printf("\n%d \t\t %d \t\t %d \t\t %d \t\t %d ",p[i],pri[i],bt[i],wt[i],tat[i]);

printf("\nAverage Waiting Time is --- %f",wtavg/n);

printf("\nAverage Turnaround Time is --- %f",tatavg/n);

}

OUTPUT:

