OPERATING SYSTEMS

Report

Assignment Simulation Based

**Student Name :** Nagamolla Sandeep Reddy.

**Student ID :** 11705798

**Student RollNo :** 23

**Emai l Address :** sr274952@gmail.com

**PROBLEM STATEMNEMT:**

Considering 4 processes with the arrival time and the burst time requirement of the processes the scheduler schedules the processes by interrupting the processor after every 4 units of time and does consider the completion of the process in this iteration. The schedulers then checks for the number of processes waiting for the processor and allots the processor to the process but interrupting the processor after every 7 units of time and considers the completion of the process in this iteration. The scheduler after the second iteration checks for the number of processes waiting for the processor and now provides the processor to the process with the least time requirement to go in the terminated state. The inputs for the number of requirements, arrival time and burst time should be provided by the user.

**CODE:**

**#include<stdio.h>**

**#include<conio.h>**

**void rr(int no,int remt[10],int Cur\_t,int arT[10], int bsT[10]);**

**main()**

**{**

**IntProc\_no,j,no,CurT,RemProc,indicator,time\_quan,wait,tut,arT[10],bsT[10],**

**remt[10],x=1;**

**indicator = 0;**

**wait = 0;**

**tut = 0;**

**printf("Enter number of processes ");**

**scanf("%d",&no);**

**RemProc = no;**

**printf("\nEnter the arrival time and burst time of the processes\n");**

**for(Proc\_no = 0;Proc\_no < no;Proc\_no++)**

**{**

**printf("\nProcess P%d\n",Proc\_no+1);**

**printf("Arrival time = ");**

**scanf("%d",&arT[Proc\_no]);**

**printf("Burst time = ");**

**scanf("%d",&bsT[Proc\_no]);**

**printf("\n");**

**remt[Proc\_no]=bsT[Proc\_no];**

**}**

**printf("The details of time quantum are as follows:\n");**

**printf("\nThe time quantum for first round is 4.\n");**

**time\_quan=4;**

**CurT=0;**

**printf("\n\nProcess\t\tTurnaround Time\t\tWaiting Time");**

**for(Proc\_no=0;RemProc!=0;)**

**{**

**if(remt[Proc\_no]<=time\_quan && remt[Proc\_no]>0)**

**{**

**CurT+=remt[Proc\_no];**

**remt[Proc\_no]=0;**

**indicator=1;**

**}**

**else if(remt[Proc\_no]>0)**

**{**

**remt[Proc\_no]-=time\_quan;**

**CurT+=time\_quan;**

**}**

**if(remt[Proc\_no]==0 && indicator==1)**

**{**

**RemProc--;**

**printf("\nP%d\t\t\t%d\t\t\t%d",Proc\_no+1,CurT-arT[Proc\_no],CurT-bsT[Proc\_no]-arT[Proc\_no]);**

**wait+=CurT-arT[Proc\_no]-bsT[Proc\_no];**

**tut+=CurT-arT[Proc\_no];**

**indicator=0;**

**}**

**if(Proc\_no==no-1){**

**x++;**

**if(x==2){**

**Proc\_no=0;**

**time\_quan=7;**

**printf("\n\nThe time quantum for second round is 7. \n");**

**}**

**else{**

**break;**

**}**

**}**

**else if(CurT >= arT[Proc\_no+1]){**

**Proc\_no++;**

**}**

**else{**

**Proc\_no=0;**

**}**

**}**

**rr(no,remt,CurT,arT,bsT);**

**return 0;**

**}**

**void rr(int no,int remt[10],int Cur\_t,int arT[10], int bsT[10]){**

**float avg\_wait,avg\_tut;**

**int i,j,n=no,temp,btime[20],Proc\_no[20],w\_time[20],tut\_t[20],total=0,loc;**

**printf("\n\nThird round with least burst time.\n");**

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

**{**

**btime[i]=remt[i];**

**w\_time[i]=Cur\_t-arT[i]-btime[i];**

**Proc\_no[i]=i+1;**

**}**

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

**{**

**loc=i;**

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

**{**

**if(btime[j]<btime[loc]){**

**loc=j;**

**}**

**}**

**temp=btime[i];**

**btime[i]=btime[loc];**

**btime[loc]=temp;**

**temp=Proc\_no[i];**

**Proc\_no[i]=Proc\_no[loc];**

**Proc\_no[loc]=temp;**

**}**

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

**{**

**for(j=0;j<i;j++){**

**w\_time[i]+=btime[j];**

**}**

**total+=w\_time[i];**

**}**

**avg\_wait=(float)total/n;**

**total=0;**

**printf("\n\nProcess\t\tBurst time\t\twaiting time\t\tTurnaround Time");**

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

**{**

**tut\_t[i]=btime[i]+w\_time[i];**

**total=total + tut\_t[i];**

**printf("\nP%d\t\t\t%d\t\t\t%d\t\t\t%d",Proc\_no[i],btime[i],w\_time[i],tut\_t[i]);**

**}**

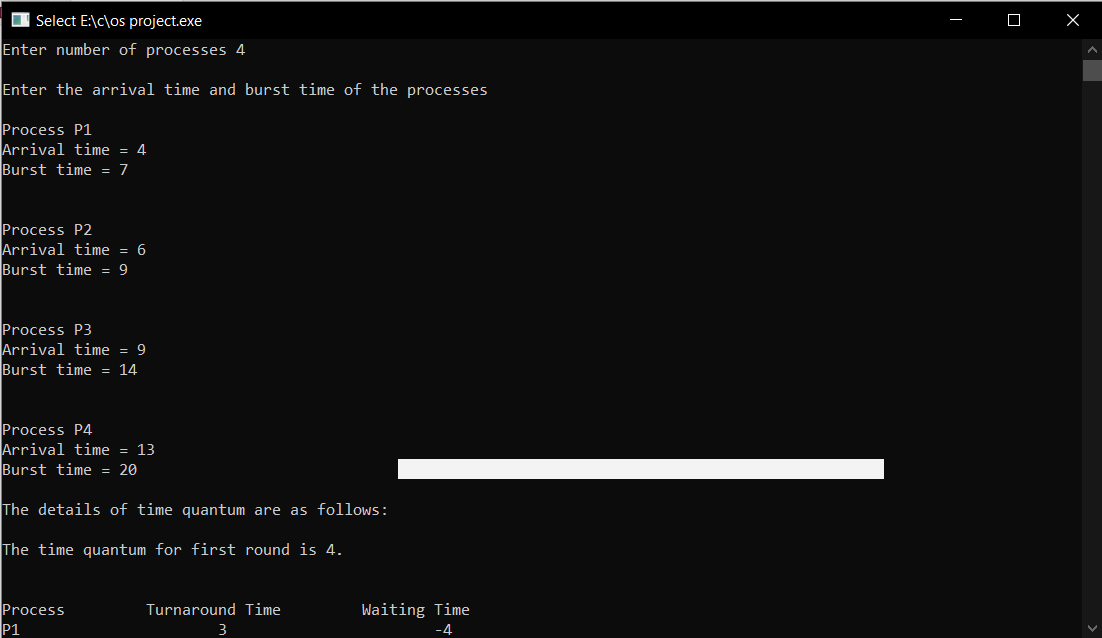
**avg\_tut=(float)total/n;**

**printf("\n\nAverage waiting time = %f",avg\_wait);**

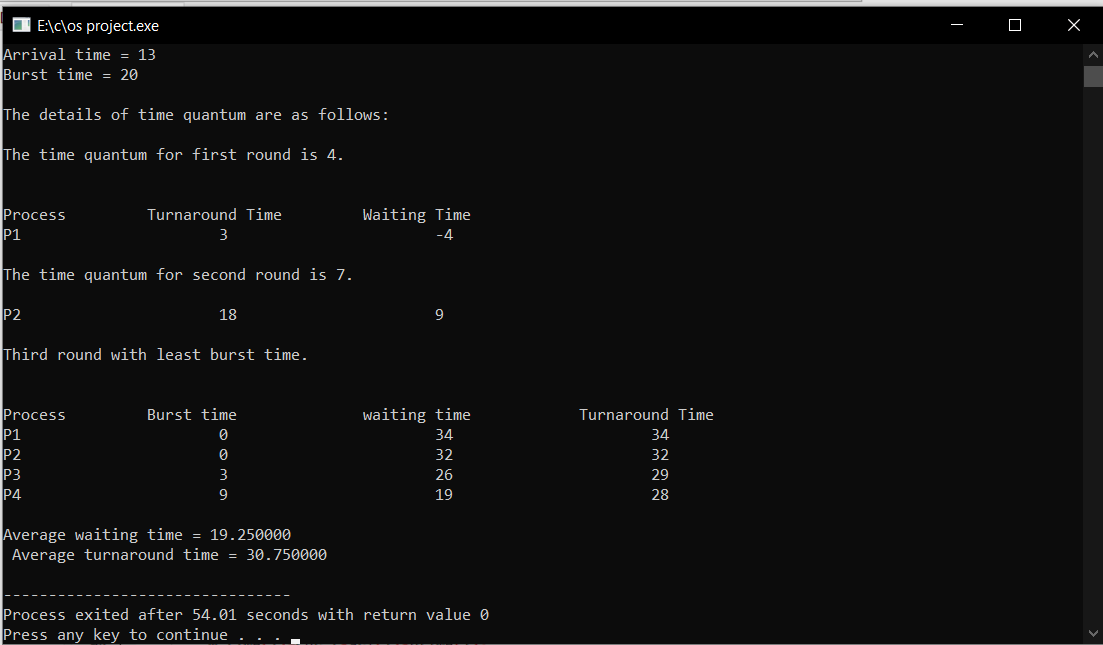
**printf("\n Average turnaround time = %f\n",avg\_tut);**

**}**

**TEST CASES:**

****

**TEST CASES:**

****

**Waiting time = Turn around time-Burst time.**

**Turn around time = completion time-Arrival time.**

**Completion time = Time at which process completes its execution.**

**Arrival time = Time at which the process arrives in the ready queue.**

**Burst time = Time required by a process for CPU execution.**

**DESCRIPTION:**

This project was done with the help of “Round robin sheduling “.In this project the user will enter the no of process along with Arrival time and Burst time.In this first we will solve by taking time quantum as”4”.If any of the process whose Burst time is less than or equal to 4 will execute at time quantum 4.For those process whose Burst time is greater than 4 will execute at time quantum 7.Next,the loop executes with least burst time and calculates the waiting time and turnaround time taken by the process to reach least burst time.