CENTRAL UNIVERSITY OF HARYANA

**Department of Computer Science & Engineering under SOET**



**Operating System**

Write a program to simulate the following preemptive/non-preemptive CPU scheduling algorithms to find turnaround time and waiting time for the above problem.

Submitted by

**Ajit Singh**

**Roll No- 192219**

Submitted to

**Dr. Benay Kumar Ray**

Assistant Professor

Central University of Haryana (SOET)

**Program - 1:** Write a program to simulate the following preemptive/ non-preemptive CPU scheduling algorithms to find turnaround time and waiting time for the above problem.

1. Round Robin
2. SJF
3. FCFS
4. Priority

**Code:**

**Round Robin**

#include<stdio.h> struct process

{

int id,AT,BT,WT,TAT;

};

struct process a[10];

int queue[100]; int front=-1; int rear=-1;

void insert(int n)

{

if(front==-1) front=0; rear=rear+1; queue[rear]=n;

}

int delete()

{

int n; n=queue[front]; front=front+1; return n;

}

int main()

{

int n,TQ,p,TIME=0;

int temp[10],exist[10]={0};

float total\_wt=0,total\_tat=0,Avg\_WT,Avg\_TAT; printf("Enter the number of the process\n"); scanf("%d",&n);

printf("Enter the arrival time and burst time of the process\n"); printf("AT BT\n");

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

{

scanf("%d%d",&a[i].AT,&a[i].BT); a[i].id=i;

temp[i]=a[i].BT;

}

printf("Enter the time quantum\n"); scanf("%d",&TQ);

insert(0); exist[0]=1;

// until ready queue is empty while(front<=rear)

{

p=delete();

if(a[p].BT>=TQ)

{

a[p].BT=a[p].BT-TQ; TIME=TIME+TQ;

}

else

{

TIME=TIME+a[p].BT; a[p].BT=0;

}

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

{

if(exist[i]==0 && a[i].AT<=TIME)

{

insert(i); exist[i]=1;

}

}

if(a[p].BT==0)

{

a[p].TAT=TIME-a[p].AT;

a[p].WT=a[p].TAT-temp[p]; total\_tat=total\_tat+a[p].TAT; total\_wt=total\_wt+a[p].WT;

}

else

{

insert(p);

}

}

Avg\_TAT=total\_tat/n; Avg\_WT=total\_wt/n;

// printing of the answer printf("ID WT TAT\n"); for(int i=0;i<n;i++)

{

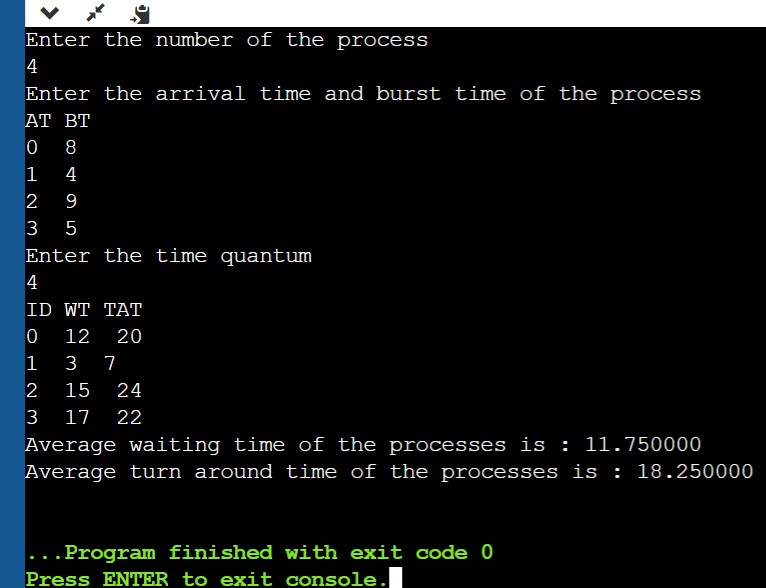
printf("%d %d %d\n",a[i].id,a[i].WT,a[i].TAT);

}

printf("Average waiting time of the processes is : %f\n",Avg\_WT); printf("Average turn around time of the processes is : %f\n",Avg\_TAT); return 0;

}

# Output:



**Code:**

**SJF (**shortest job first scheduling**) :**

#include <iostream>

using namespace std;

int mat[10][6];

void swap(int\* a, int\* b)

{

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void arrangeArrival(int num, int mat[][6])

{

for (int i = 0; i < num; i++) {

for (int j = 0; j < num - i - 1; j++) {

if (mat[j][1] > mat[j + 1][1]) {

for (int k = 0; k < 5; k++) {

swap(mat[j][k], mat[j + 1][k]);

}

}

}

}

}

void completionTime(int num, int mat[][6])

{

int temp, val;

mat[0][3] = mat[0][1] + mat[0][2];

mat[0][5] = mat[0][3] - mat[0][1];

mat[0][4] = mat[0][5] - mat[0][2];

for (int i = 1; i < num; i++) {

temp = mat[i - 1][3];

int low = mat[i][2];

for (int j = i; j < num; j++) {

if (temp >= mat[j][1] && low >= mat[j][2]) {

low = mat[j][2];

val = j;

}

}

mat[val][3] = temp + mat[val][2];

mat[val][5] = mat[val][3] - mat[val][1];

mat[val][4] = mat[val][5] - mat[val][2];

for (int k = 0; k < 6; k++) {

swap(mat[val][k], mat[i][k]);

}

}

}

int main()

{

int num, temp;

cout << "Enter number of Process: ";

cin >> num;

cout << "...Enter the process ID...\n";

for (int i = 0; i < num; i++) {

cout << "...Process " << i + 1 << "...\n";

cout << "Enter Process Id: ";

cin >> mat[i][0];

cout << "Enter Arrival Time: ";

cin >> mat[i][1];

cout << "Enter Burst Time: ";

cin >> mat[i][2];

}

cout << "Before Arrange...\n";

cout << "Process ID\tArrival Time\tBurst Time\n";

for (int i = 0; i < num; i++) {

cout << mat[i][0] << "\t\t" << mat[i][1] << "\t\t"

<< mat[i][2] << "\n";

}

arrangeArrival(num, mat);

completionTime(num, mat);

cout << "Final Result...\n";

cout << "Process ID\tArrival Time\tBurst Time\tWaiting "

"Time\tTurnaround Time\n";

for (int i = 0; i < num; i++) {

cout << mat[i][0] << "\t\t" << mat[i][1] << "\t\t"

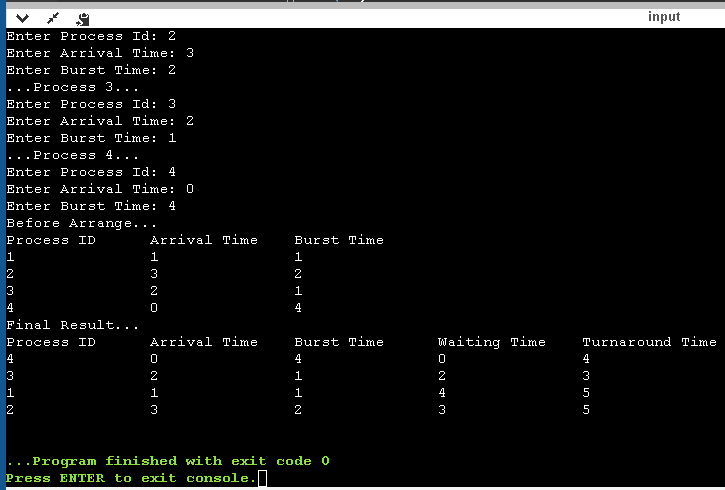
<< mat[i][2] << "\t\t" << mat[i][4] << "\t\t"

<< mat[i][5] << "\n";

}

}

# Output:

****

**Code:**

**SJF/SRTF: Preemptive**

#include <stdio.h>

int main()

{

int arrival\_time[10], burst\_time[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, turnaround\_time = 0, end;

float average\_waiting\_time, average\_turnaround\_time;

printf("\nEnter the Total Number of Processes:\t");

scanf("%d", &limit);

printf("\nEnter Details of %d Processesn", limit);

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

{

printf("\nEnter Arrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Enter Burst Time:\t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

burst\_time[9] = 9999;

for(time = 0; count != limit; time++)

{

smallest = 9;

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

{

if(arrival\_time[i] <= time && burst\_time[i] < burst\_time[smallest] && burst\_time[i] > 0)

{

smallest = i;

}

}

burst\_time[smallest]--;

if(burst\_time[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - arrival\_time[smallest] - temp[smallest];

turnaround\_time = turnaround\_time + end - arrival\_time[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

average\_turnaround\_time = turnaround\_time / limit;

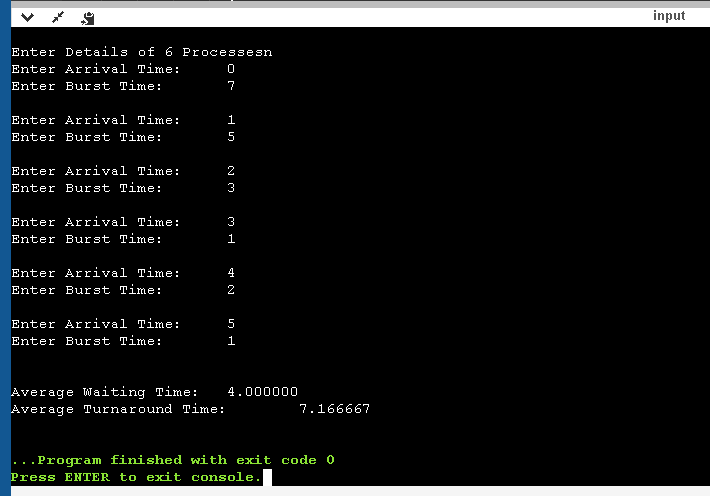
printf("\n\nAverage Waiting Time:\t%lf\n", average\_waiting\_time);

printf("Average Turnaround Time:\t%lf\n", average\_turnaround\_time);

return 0;

}

**Output:**

****

**Code:**

**FCFS (**First Come First Served**)**

#include<iostream>

#include <algorithm>

#include<iomanip>

using namespace std;

struct process\_struct

{

int pid;

int at;

int bt;

int ct,wt,tat,rt,start\_time;

}ps[100];

bool comparatorAT(struct process\_struct a,struct process\_struct b)

{

int x =a.at;

int y =b.at;

return x < y;

}

bool comparatorPID(struct process\_struct a,struct process\_struct b)

{

int x =a.pid;

int y =b.pid;

return x < y;

}

int main()

{

int n;

cout<<"Enter total number of processes: ";

cin>>n;

float sum\_tat=0,sum\_wt=0,sum\_rt=0;

int length\_cycle,total\_idle\_time=0;

float cpu\_utilization;

cout << fixed << setprecision(2);

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

{

cout<<"\nEnter Process" <<i<< "Arrival Time: ";

cin >> ps[i].at;

ps[i].pid=i;

}

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

{

cout<<"\nEnter Process" <<i<< "Burst Time: ";

cin >> ps[i].bt;

}

//sort

sort(ps,ps+n, comparatorAT);

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

{

ps[i].start\_time = (i==0) ? ps[i].at : max(ps[i].at, ps[i-1].ct);

ps[i].ct = ps[i].start\_time + ps[i].bt;

ps[i].tat = ps[i].ct-ps[i].at;

ps[i].wt = ps[i].tat-ps[i].bt;

ps[i].rt=ps[i].wt;

sum\_tat += ps[i].tat;

sum\_wt += ps[i].wt;

sum\_rt += ps[i].rt;

total\_idle\_time += (i==0) ? 0 : (ps[i].start\_time - ps[i-1].ct);

}

length\_cycle = ps[n-1].ct - ps[0].start\_time;

sort(ps,ps+n, comparatorPID);

cout<<"\nProcess No.\tAT\tCPU Burst Time\tCT\tTAT\tWT\tRT\n";

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

cout<<i<<"\t\t"<<ps[i].at<<"\t"<<ps[i].bt<<"\t\t"<<ps[i].ct<<"\t"<<ps[i].tat<<"\t"<<ps[i].wt<<"\t"<<ps[i].rt<<endl;

cout<<endl;

cpu\_utilization = (float)(length\_cycle - total\_idle\_time)/ length\_cycle;

cout<<"\nAverage Turn Around time= "<< sum\_tat/n;

cout<<"\nAverage Waiting Time= "<<sum\_wt/n;

cout<<"\nAverage Response Time= "<<sum\_rt/n;

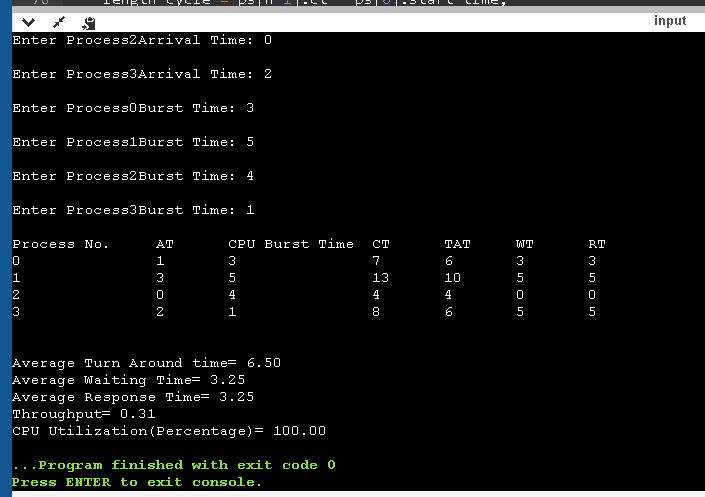
cout<<"\nThroughput= "<<n/(float)length\_cycle;

cout<<"\nCPU Utilization(Percentage)= " << cpu\_utilization\*100;

return 0;

# }

# Output:

****

**Code:**

**Priority (Non-preemptive)**

#include<stdio.h>

struct process

{

int WT,AT,BT,TAT,PT;

};

struct process a[10];

int main()

{

int n,temp[10],t,count=0,short\_p;

float total\_WT=0,total\_TAT=0,Avg\_WT,Avg\_TAT;

printf("Enter the number of the process\n");

scanf("%d",&n);

printf("Enter the arrival time , burst time and priority of the process\n");

printf("AT BT PT\n");

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

{

scanf("%d%d%d",&a[i].AT,&a[i].BT,&a[i].PT);

temp[i]=a[i].BT;

}

a[9].PT=10000;

for(t=0;count!=n;t++)

{

short\_p=9;

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

{

if(a[short\_p].PT>a[i].PT && a[i].AT<=t && a[i].BT>0)

{

short\_p=i;

}

}

a[short\_p].BT=a[short\_p].BT-1;

if(a[short\_p].BT==0)

{

count++;

a[short\_p].WT=t+1-a[short\_p].AT-temp[short\_p];

a[short\_p].TAT=t+1-a[short\_p].AT;

total\_WT=total\_WT+a[short\_p].WT;

total\_TAT=total\_TAT+a[short\_p].TAT;

}

}

Avg\_WT=total\_WT/n;

Avg\_TAT=total\_TAT/n;

printf("ID WT TAT\n");

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

{

printf("%d %d\t%d\n",i+1,a[i].WT,a[i].TAT);

}

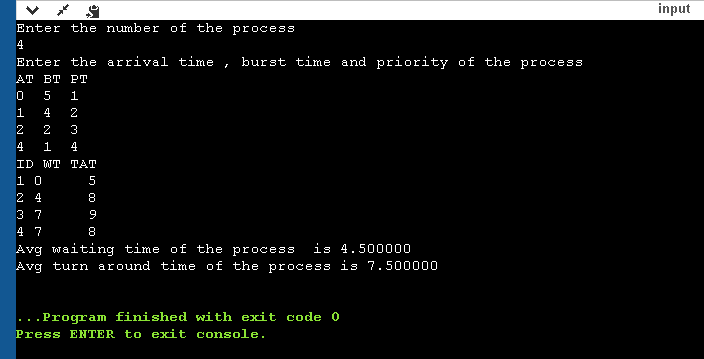
printf("Avg waiting time of the process is %f\n",Avg\_WT);

printf("Avg turn around time of the process is %f\n",Avg\_TAT);

return 0;

}

**Output:**

****

**Code:**

**Priority(Preemptive)**

#include<stdio.h>

struct process

{

int WT,AT,BT,TAT,PT;

};

struct process a[10];

int main()

{

int n,temp[10],t,count=0,short\_p;

float total\_WT=0,total\_TAT=0,Avg\_WT,Avg\_TAT;

printf("Enter the number of the process\n");

scanf("%d",&n);

printf("Enter the arrival time , burst time and priority of the process\n");

printf("AT BT PT\n");

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

{

scanf("%d%d%d",&a[i].AT,&a[i].BT,&a[i].PT);

temp[i]=a[i].BT;

}

a[9].PT=10000;

for(t=0;count!=n;t++)

{

short\_p=9;

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

{

if(a[short\_p].PT>a[i].PT && a[i].AT<=t && a[i].BT>0)

{

short\_p=i;

}

}

a[short\_p].BT=a[short\_p].BT-1;

if(a[short\_p].BT==0)

{

count++;

a[short\_p].WT=t+1-a[short\_p].AT-temp[short\_p];

a[short\_p].TAT=t+1-a[short\_p].AT;

total\_WT=total\_WT+a[short\_p].WT;

total\_TAT=total\_TAT+a[short\_p].TAT;

}

}

Avg\_WT=total\_WT/n;

Avg\_TAT=total\_TAT/n;

printf("ID WT TAT\n");

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

{

printf("%d %d\t%d\n",i+1,a[i].WT,a[i].TAT);

}

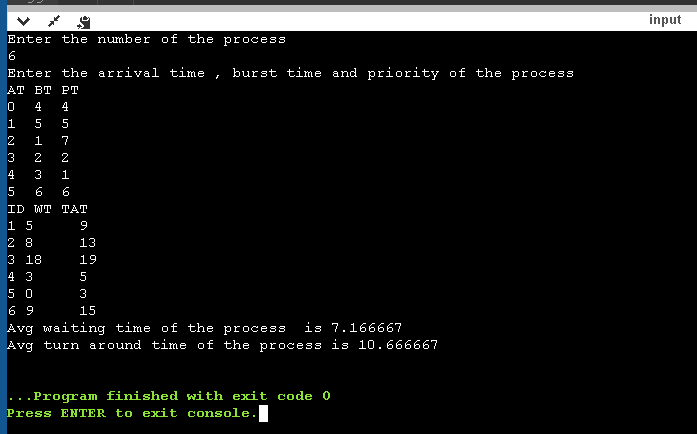
printf("Avg waiting time of the process is %f\n",Avg\_WT);

printf("Avg turn around time of the process is %f\n",Avg\_TAT);

return 0;

}

**Output:**

****