\*

\* C Program to Implement non preemptive SJF Scheduling

\*/

#include<stdio.h>

int main()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,totalT=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

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

{

printf("p%d:",i+1);

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

p[i]=i+1;

}

//sorting of burst times

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

{

pos=i;

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

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

//finding the waiting time of all the processes

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

{

wt[i]=0;

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

//individual WT by adding BT of all previous completed processes

wt[i]+=bt[j];

//total waiting time

total+=wt[i];

}

//average waiting time

avg\_wt=(float)total/n;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

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

{

//turnaround time of individual processes

tat[i]=bt[i]+wt[i];

//total turnaround time

totalT+=tat[i];

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

}

//average turnaround time

avg\_tat=(float)totalT/n;

printf("\n\nAverage Waiting Time=%f",avg\_wt);

printf("\nAverage Turnaround Time=%f",avg\_tat);

}

Output -

Enter number of process:4

Enter Burst Time:

p1:5

p2:4

p3:12

p4:7

Process Burst Time Waiting Time Turnaround Time

p2 4 0 4

p1 5 4 9

p4 7 9 16

p3 12 16 28

Average Waiting Time=7.250000

Average Turnaround Time=14.250000

/\*

\* FCFS Scheduling Program in C

\*/

#include <stdio.h>

int main()

{

int pid[15];

int bt[15];

int n;

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

scanf("%d",&n);

printf("Enter process id of all the processes: ");

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

{

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

}

printf("Enter burst time of all the processes: ");

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

{

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

}

int i, wt[n];

wt[0]=0;

//for calculating waiting time of each process

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

{

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

}

printf("Process ID Burst Time Waiting Time TurnAround Time\n");

float twt=0.0;

float tat= 0.0;

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

{

printf("%d\t\t", pid[i]);

printf("%d\t\t", bt[i]);

printf("%d\t\t", wt[i]);

//calculating and printing turnaround time of each process

printf("%d\t\t", bt[i]+wt[i]);

printf("\n");

//for calculating total waiting time

twt += wt[i];

//for calculating total turnaround time

tat += (wt[i]+bt[i]);

}

float att,awt;

//for calculating average waiting time

awt = twt/n;

//for calculating average turnaround time

att = tat/n;

printf("Avg. waiting time= %f\n",awt);

printf("Avg. turnaround time= %f",att);

}

Output -

Enter the number of processes: 3

Enter process id of all the processes: 1 2 3

Enter burst time of all the processes: 5 11 11

Process ID Burst Time Waiting Time TurnAround Time

1 5 0 5

2 11 5 16

3 11 16 27

Avg. waiting time= 7.000000

Avg. turnaround time= 16.000000

/\*

\* Round Robin Scheduling Program in C

\*/

#include<stdio.h>

int main()

{

//Input no of processed

int n;

printf("Enter Total Number of Processes:");

scanf("%d", &n);

int wait\_time = 0, ta\_time = 0, arr\_time[n], burst\_time[n], temp\_burst\_time[n];

int x = n;

//Input details of processes

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

{

printf("Enter Details of Process %d \n", i + 1);

printf("Arrival Time: ");

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

printf("Burst Time: ");

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

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

}

//Input time slot

int time\_slot;

printf("Enter Time Slot:");

scanf("%d", &time\_slot);

//Total indicates total time

//counter indicates which process is executed

int total = 0, counter = 0,i;

printf("Process ID Burst Time Turnaround Time Waiting Time\n");

for(total=0, i = 0; x!=0; )

{

// define the conditions

if(temp\_burst\_time[i] <= time\_slot && temp\_burst\_time[i] > 0)

{

total = total + temp\_burst\_time[i];

temp\_burst\_time[i] = 0;

counter=1;

}

else if(temp\_burst\_time[i] > 0)

{

temp\_burst\_time[i] = temp\_burst\_time[i] - time\_slot;

total += time\_slot;

}

if(temp\_burst\_time[i]==0 && counter==1)

{

x--; //decrement the process no.

printf("\nProcess No %d \t\t %d\t\t\t\t %d\t\t\t %d", i+1, burst\_time[i],

total-arr\_time[i], total-arr\_time[i]-burst\_time[i]);

wait\_time = wait\_time+total-arr\_time[i]-burst\_time[i];

ta\_time += total -arr\_time[i];

counter =0;

}

if(i==n-1)

{

i=0;

}

else if(arr\_time[i+1]<=total)

{

i++;

}

else

{

i=0;

}

}

float average\_wait\_time = wait\_time \* 1.0 / n;

float average\_turnaround\_time = ta\_time \* 1.0 / n;

printf("\nAverage Waiting Time:%f", average\_wait\_time);

printf("\nAvg Turnaround Time:%f", average\_turnaround\_time);

return 0;

}

Output -

Enter Total Number of Processes:3

Enter Details of Process 1

Arrival Time: 0

Burst Time: 10

Enter Details of Process 2

Arrival Time: 1

Burst Time: 8

Enter Details of Process 3

Arrival Time: 2

Burst Time: 7

Enter Time Slot:5

Process ID Burst Time Turnaround Time Waiting Time

Process No 1 10 20 10

Process No 2 8 22 14

Process No 3 7 23 16

Average Waiting Time: 13.333333

Avg Turnaround Time: 21.666666

/\*

\* C program to implement priority scheduling

\*/

#include <stdio.h>

//Function to swap two variables

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

{

int temp=\*a;

\*a=\*b;

\*b=temp;

}

int main()

{

int n;

printf("Enter Number of Processes: ");

scanf("%d",&n);

// b is array for burst time, p for priority and index for process id

int b[n],p[n],index[n];

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

{

printf("Enter Burst Time and Priority Value for Process %d: ",i+1);

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

index[i]=i+1;

}

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

{

int a=p[i],m=i;

//Finding out highest priority element and placing it at its desired position

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

{

if(p[j] > a)

{

a=p[j];

m=j;

}

}

//Swapping processes

swap(&p[i], &p[m]);

swap(&b[i], &b[m]);

swap(&index[i],&index[m]);

}

// T stores the starting time of process

int t=0;

//Printing scheduled process

printf("Order of process Execution is\n");

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

{

printf("P%d is executed from %d to %d\n",index[i],t,t+b[i]);

t+=b[i];

}

printf("\n");

printf("Process Id Burst Time Wait Time TurnAround Time\n");

int wait\_time=0;

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

{

printf("P%d %d %d %d\n",index[i],b[i],wait\_time,wait\_time + b[i]);

wait\_time += b[i];

}

return 0;

}

Output -

Enter Number of Processes: 3

Enter Burst Time and Priority Value for Process 1: 10 2

Enter Burst Time and Priority Value for Process 2: 5 0

Enter Burst Time and Priority Value for Process 3: 8 1

Order of process Execution is

P1 is executed from 0 to 10

P3 is executed from 10 to 18

P2 is executed from 18 to 23

Process Id Burst Time Wait Time TurnAround Time

P1 10 0 10

P3 8 10 18

P2 5 18 23