## PROGRAM NO-1

## //Program to show First Come First Served Scheduling algorithm.

#include<stdio.h>

#include<conio.h>

void main()

{

int burst\_time[100],gant\_chart[100],arrival\_time[100];

int i,t,j,len;

float wt=0,ta=0,s=0;

clrscr();

printf("\t\tWelcome to the c program.\n\n");

printf("Program for first come first served sheduling.\n");

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

scanf("%d",&len);

printf("Enter the burst time.\n");

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

{

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

}

printf("Enter the arrival time.\n");

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

{

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

s+=arrival\_time[i];

}

printf("Process\t\tBurst Time\t\tArrival Time\n");

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

{ printf("P%d\t\t",i+1);

printf("%d\t\t\t%d\n",burst\_time[i],arrival\_time[i]);

}

gant\_chart[0]=0;

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

{

gant\_chart[i]=(gant\_chart[i-1]+burst\_time[i-1]);

}

printf("\nThe gant chart is:-\n");

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

printf("|%d|\t",gant\_chart[i]);

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

ta+=gant\_chart[i];

wt=ta-gant\_chart[len];

printf("\nTurn around time=%f\t",ta-s);

printf("\nAverage turn around time=%f\t",(ta-s)/len);

printf("\nWaiting time=%f\t",wt-s);

printf("\nAverage waiting time=%f\t",(wt-s)/len);

getch();}

## OUTPUT:-

Welcome to the c program.

Program for first come first serve sheduling.

Enter the number of process

3

Enter the burst time.

20

7

5

Enter the arrival time.

0

1

2

Process Burst Time Arrival Time

P1 20 0

P2 7 1

P3 5 2

The gant chart is:-

|0| |20| |27| |32|

Turn around time=76.000000

Average turn around time=25.333333

Waiting time=44.000000

Average waiting time=14.666667

## PROGRAM NO-2

## //Program to show Shortest Job First Scheduling algorithm.

#include<stdio.h>

#include<conio.h>

void main()

{

int burst\_time[100],gant\_chart[100];

int i,t,j,len;

float wt=0,ta=0,rt=0;

clrscr();

printf("\t\tWelcome to the c program.\n\n");

printf("Program for shortest job first sheduling.\n");

printf("Using non preemptive.\n");

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

scanf("%d",&len);

printf("Enter the burst time.\n");

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

{

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

}

printf("Process\t\tBurst Time.\n");

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

{ printf("P%d\t\t",i+1);

printf("%d\t\t\n",burst\_time[i]);

}

for(i=1;i<3;i++)

{

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

{

if(burst\_time[j]>burst\_time[j+1])

{

t=burst\_time[j];

burst\_time[j]=burst\_time[j+1];

burst\_time[j+1]=t;

}

}

}

gant\_chart[0]=0;

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

{

gant\_chart[i]=(gant\_chart[i-1]+burst\_time[i-1]);

}

printf("\nThe gant chart is:-\n\n");

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

printf("|%d|\t",gant\_chart[i]);

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

{

ta+=gant\_chart[i];

}

wt=ta-gant\_chart[len];

printf("\nTurn around time=%f\n",ta);

printf("Average turn around time=%f\n",ta/len);

printf("Waiting time=%f\n",wt-s);

printf("Average waiting time=%f\n",wt/len);

getch();

}

## OUTPUT:-

Welcome to the c program.

Program for shortest job first sheduling.

Using non preemptive.

Enter the number of process

3

Enter the burst time.

20

7

5

Process Burst Time.

P1 20

P2 7

P3 5

The gant chart is:-

|0| |5| |12| |32|

Turn around time=49.000000

Average turn around time=16.333333

Waiting time=17.000000

Average waiting time=5.666667

## PROGRAM NO-4

//Program to implement Priority scheduling algorithm.

#include<stdio.h>

#include<conio.h>

void main()

{

int burst\_time[100],gant\_chart[100],priority[100];

int i,t,j,len;

float wt=0,ta=0,rt=0;

clrscr();

printf("\t\tWelcome to the c program.\n\n");

printf("Program for priority sheduling.\n");

printf("Using non preemptive.\n");

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

scanf("%d",&len);

printf("Enter the burst time.\n");

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

{

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

}

printf("Enter the priority.\n");

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

{

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

}

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

{

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

{

if(priority[j]>priority[j+1])

{

t=priority[j];

priority[j]=priority[j+1];

priority[j+1]=t;

t=burst\_time[j];

burst\_time[j]=burst\_time[j+1];

burst\_time[j+1]=t;

}

}

}

printf("Process\t\tBurst Time.\t\tPriority.\n");

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

{ printf("P%d\t\t",i+1);

printf("%d\t\t%d\t\t\n",burst\_time[i],priority[i]);

}

gant\_chart[0]=0;

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

{

gant\_chart[i]=(gant\_chart[i-1]+burst\_time[i-1]);

}

printf("\nThe gant chart is:-\n\n");

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

printf("|%d|\t",gant\_chart[i]);

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

{

ta+=gant\_chart[i];

}

wt=ta-gant\_chart[len];

printf("\nTotle turn around time=%f\n",ta);

printf("Average turn around time=%f\n",ta/len);

printf("Totle waiting time=%f\n",wt);

printf("Average waiting time=%f\n",wt/len);

getch();

}

## OUTPUT:-

Welcome to the c program.

Program for priority sheduling.

Using non preemptive.

Enter the number of process

5

Enter the burst time.

10 1 2 1 5

Enter the priority.

3 1 4 5 2

Process Burst Time. Priority.

P1 1 1

P2 5 2

P3 10 3

P4 2 4

P5 1 5

The gant chart is:-

|0| |1| |6| |16| |18| |19|

Totle turn around time=60.000000

Average turn around time=12.000000

Totle waiting time=41.000000

Average waiting time=8.200000

## PROGRAM NO-6

## //Program to implement Bankers algorithm.

#include<stdio.h>

#include<conio.h>

struct temp

{

int value;

int status;

};

void main()

{

struct temp need[5000];

int allocation[5000];

int max[5000];

int i,available=332,k=0,false=0,true=1;

int stack[5000],count=0,len;

clrscr();

printf("\t\tWelcome to the c program.\n");

printf("How many process?\n");

scanf("%d",&len);

printf("Enter the content of the allocation matrix.\n");

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

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

printf("Enter the content of the max matrix.\n");

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

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

printf("Enter the available content.\n");

scanf("%d",&available);

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

need[i].status=0;

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

need[i].value=max[i]-allocation[i];

printf("Process\t\tAllocation\tMax\tNeed\n");

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

printf("%d\t\t%d\t\t%d\t%d\n",i,allocation[i],max[i],need[i]);

do

{

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

{

if(need[i].status==false)

{

if(need[i].value<available)

{

available+=allocation[i];

need[i].status=true;

stack[count++]=i;

if(count==len)

goto abc;

}}

}

k++;

true=0;

}while(k<5000);

abc:

if(count==0)

printf("Sorry!No safe sequence.");

else

{

printf("The safe seqence is:-\n<");

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

printf("P%d,",stack[i]);

printf(">");

} getch(); }

## OUTPUT:-

## RUN:-1

Welcome to the c program.

How many process?

3

Enter the content of the allocation matrix.

10 20 30

Enter the content of the max matrix.

20 25 35

Enter the available content. 1

Process Allocation Max Need

0 10 20 10

1 20 25 5

2 30 35 5

Sorry!No safe sequence.

## RUN:-2

Welcome to the c program.

How many process?

5

Enter the content of the allocation matrix.

010 200 302 211 002

Enter the content of the max matrix.

753 322 902 222 433

Enter the available content.

332

Process Allocation Max Need

0 10 753 743

1 200 322 122

2 302 902 600

3 211 222 11

4 2 433 431

The safe seqence is:-

<P1,P3,P4,P0,P2,>