**ALGORITHM FOR DEADLOCK DETECTION**

**Ex.No:9**

# AIM:

To write a C program to implement algorithm for deadlock detection.

# ALGORITHM:

Step-1: Start the program.

Step-2: Declare the memory for the process.

Step-3: Read the number of process, resources, allocation matrix and available matrix. Step-4: Compare each and every process using the banker‟s algorithm.

Step-5: If the process is in safe state then it is a not a deadlock process otherwise it is a deadlock process

Step-6: produce the result of state of process Step-7: Stop the program

# PROGRAM:

#include<stdio.h> #include<conio.h> int max[100][100]; int alloc[100][100]; int need[100][100]; int avail[100];

int n,r;

void input(); void show(); void cal(); int main()

{

int i,j;

printf("\*\*\*\*\*\*\*\*\*\* Deadlock Detection Algo \*\*\*\*\*\*\*\*\*\*\*\*\n"); input();

show();

cal();

getch(); return 0;

}

void input()

{int i,j;

printf("Enter the no of Processes\t");

 scanf("%d",&n);

printf("Enter the no of resource instances\t"); scanf("%d",&r);

printf("Enter the Max Matrix\n"); for(i=0;i<n;i++)

{for(j=0;j<r;j++)

{

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

}}

printf("Enter the Allocation Matrix\n"); for(i=0;i<n;i++)

{for(j=0;j<r;j++)

{

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

}}

printf("Enter the available Resources\n"); for(j=0;j<r;j++)

{

scanf("%d",&avail[j]);

}}

void show()

{

int i,j;

printf("Process\t Allocation\t Max\t Available\t"); for(i=0;i<n;i++)

{

printf("\nP%d\t ",i+1); for(j=0;j<r;j++)

{

printf("%d ",alloc[i][j]);

}

printf("\t"); for(j=0;j<r;j++)

{printf("%d ",max[i][j]);

}

printf("\t"); if(i==0)

{

for(j=0;j<r;j++) printf("%d ",avail[j]);

}}}

void cal()

{ int finish[100],temp,need[100][100],flag=1,k,c1=0; int dead[100];

int safe[100]; int i,j;

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

{finish[i]=0;

}

//find need matrix for(i=0;i<n;i++)

{for(j=0;j<r;j++)

{

need[i][j]=max[i][j]-alloc[i][j];

}}

while(flag)

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

{int c=0; for(j=0;j<r;j++)

{if((finish[i]==0)&&(need[i][j]<=avail[j]))

{c++;

if(c==r)

{

for(k=0;k<r;k++)

{avail[k]+=alloc[i][j]; finish[i]=1;

flag=1;

}//printf("\nP%d",i); if(finish[i]==1)

{i=n;

}}}}}} j=0;

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

{

if(finish[i]==0)

{dead[j]=i; j++;

flag=1;

}}

if(flag==1)

{

printf("\n\nSystem is in Deadlock and the Deadlock process are\n"); for(i=0;i<n;i++)

{printf("P%d\t",dead[i]);

}}

else

{

printf("\nNo Deadlock Occur"); }}