Experiment Number:

Problem Statement: Deadlock – Banker’s algorithm

NAME: Manomay Jamble ROLLNO: 49

CLASS: TY(AIDS) BATCH: 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PROGARM:

#include<stdio.h>

#include<stdlib.h>

int max[10][10], allocation[10][10], need[10][10];

int available[10];

int process, resources;

void read\_matrix(int matrix[10][10]);

void display\_matrix(int matrix[10][10]);

void calculate\_need();

void banker\_algorithm();

int main(){

int i;

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

scanf("%d",&process);

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

scanf("%d",&resources);

printf("Enter the number of allocated resources\n");

read\_matrix(allocation);

printf("Enter the number of maximum resources\n");

read\_matrix(max);

printf("Enter the number of available resources\n");

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

{

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

}

printf("\nEntered data is :-\n");

printf("\nAllocated resources are: -\n");

display\_matrix(allocation);

printf("\nMaximum resources are:-\n");

display\_matrix(max);

printf("\nAvailable resources are:-\n");

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

{

printf("%d ",available[i]);

}

calculate\_need();

printf("\nNeeded resources are\n");

display\_matrix(need);

banker\_algorithm();

return 0;

}

void read\_matrix(int matrix[10][10]){

int i,j;

for(i = 0;i<process;i++){

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

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

}

}

}

void display\_matrix(int matrix[10][10]){

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

{

printf("\n ");

for (int j = 0; j < resources; j++)

{

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

}

}

}

void calculate\_need(){

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

{

for (int j = 0; j < resources; j++)

{

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

}

}

}

void banker\_algorithm(){

int i,j,k=0, flag;

int finish[10];

int safe\_seq[10];

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

{

finish[i]=0;

}

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

{

flag = 0;

if(finish[i]==0){

for (int j = 0; j < resources; j++)

{

if(need[i][j]>available[j]){

flag=1;

break;

}

}

if (flag==0)

{

finish[i]=1;

safe\_seq[k]=i;

k++;

for (int j = 0; j < resources; j++)

{

available[j]+=allocation[i][j];

}

i = -1;

}

}

}

flag=0;

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

{

if (finish[i]==0)

{

printf("The system is in deadlock\n");

flag=1;

break;

}

}

if (flag==0)

{

printf("\nThe system is in safe state\n");

printf("Safe sequence is: -\n");

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

{

printf(" P%d", safe\_seq[i]);

}

}

}

**OUTPUT:**

