OS LAB

DEADLOCK DETECTION

Name: Vishwaraj Ingawale

Roll No: 33

Class: SY-CS-B

PRN:12211415

**CODE:**

#include <stdio.h>

#include <stdlib.h>

static int mark[20];

int i, j, np, nr;

int main()

{

    int \*\*alloc, \*\*request, \*avail, \*r, \*f;

    printf("\nEnter the no of the process: ");

    scanf("%d", &np);

    printf("\nEnter the no of resources: ");

    scanf("%d", &nr);

    alloc = (int \*\*)malloc(np \* sizeof(int \*));

    request = (int \*\*)malloc(np \* sizeof(int \*));

    avail = (int \*)malloc(nr \* sizeof(int));

    r = (int \*)malloc(nr \* sizeof(int));

    f = (int \*)malloc(np \* sizeof(int));

    for(i = 0;i < np;i ++)

        f[i] = 0;

    for (i = 0; i < nr; i++)

    {

        printf("\nTotal Amount of the Resource R % d: ", i + 1);

        scanf("%d", &r[i]);

    }

    printf("\nEnter the request matrix:");

    for (i = 0; i < np; i++)

    {

        request[i] = (int \*)malloc(nr \* sizeof(int));

        for (j = 0; j < nr; j++)

        {

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

        }

    }

    printf("\nEnter the allocation matrix:");

    for (i = 0; i < np; i++)

    {

        alloc[i] = (int \*)malloc(nr \* sizeof(int));

        for (j = 0; j < nr; j++)

        {

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

        }

    }

    /\*Available Resource calculation\*/

    for (j = 0; j < nr; j++)

    {

        avail[j] = r[j];

        for (i = 0; i < np; i++)

        {

            avail[j] -= alloc[i][j];

        }

    }

    for (i = 0; i < nr; i++)

    {

        printf("Resc %d :: %d", i, avail[i]);

    }

    while (1)

    {

        int deadlock = 1;

        for (i = 0; i < np ;i++){

            int canBeProc = 1;

            for (j = 0; j < nr; j++){

                if (request[i][j] > avail[j]){

                    canBeProc = 0;

                }

            }

            if (canBeProc){

                deadlock = 0;

                f[i] = 1;

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

                    avail[j] += alloc[i][j];

                }

            }

        }

        if (deadlock){

            printf("\n Deadlock detected");

            break;

        }

        int completed = 1;

        for (i = 0; i < np ; i++){

            if (f[i] == 0){

                completed = 0;

            }

        }

        if (completed){

            printf("\n Deadlock not detected");

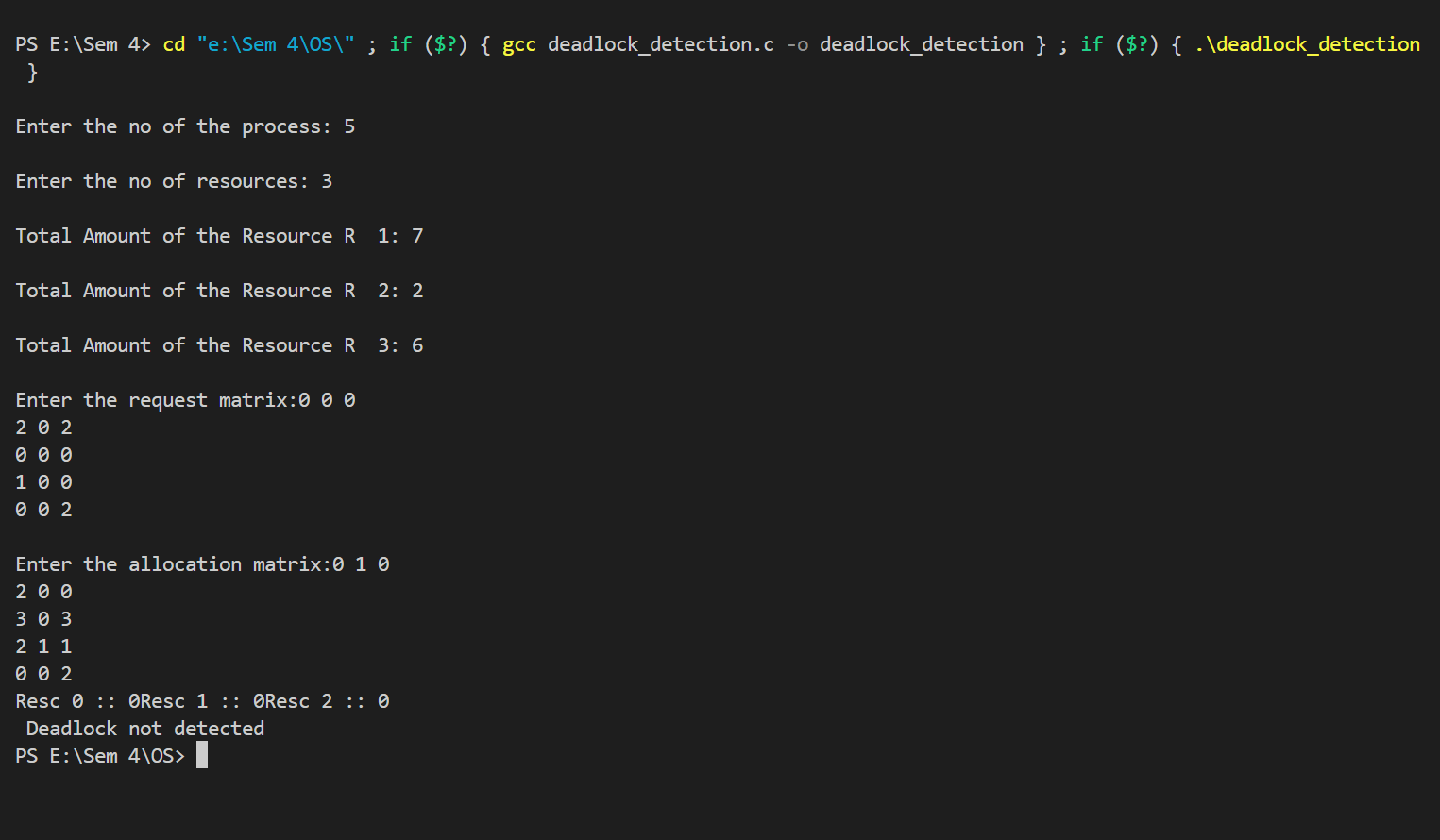
            break;

        }

    }

}

**OUTPUT:**

****