# **ASG 09**

**Name:** Yatharth Thakare

**Roll No:** 51

**PRN:** 12111403

**Subject:** OS

**Title: Banker’s Algorithm**

**Code:**

#include <bits/stdc++.h>

using namespace std;

#define MAX\_PROCESSES 10

#define MAX\_RESOURCES 10

int isSafe(int processes[], int available[], int max[][MAX\_RESOURCES], int alloc[][MAX\_RESOURCES], int need[][MAX\_RESOURCES], int n, int m, int safeSeq[])

{

int work[MAX\_RESOURCES];

int finish[MAX\_PROCESSES] = {0};

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

work[i] = available[i];

int count = 0;

while (count < n)

{

int found = 0;

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

{

if (finish[i] == 0)

{

int j;

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

{

if (need[i][j] > work[j])

break;

}

if (j == m)

{

for (int x = 0; x < m; x++)

work[x] += alloc[i][x];

finish[i] = 1;

safeSeq[count] = i;

count++;

found = 1;

}

}

}

if (!found)

return 0;

}

return 1;

}

int main()

{

int n, m;

int processes[MAX\_PROCESSES];

int available[MAX\_RESOURCES];

int max[MAX\_PROCESSES][MAX\_RESOURCES];

int alloc[MAX\_PROCESSES][MAX\_RESOURCES];

int need[MAX\_PROCESSES][MAX\_RESOURCES];

int safeSeq[MAX\_PROCESSES];

cout << "Enter the number of processes: ";

cin >> n;

cout << "Enter the number of resources: ";

cin >> m;

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

processes[i] = i;

cout << "Enter the available resources: ";

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

cin >> available[i];

cout << "Enter the maximum demand matrix:\n";

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

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

cin >> max[i][j];

cout << "Enter the allocation matrix:\n";

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

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

cin >> alloc[i][j];

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

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

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

if (isSafe(processes, available, max, alloc, need, n, m, safeSeq))

{

cout << "System is in a safe state.\nSafe sequence: ";

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

{

cout << processes[safeSeq[i]];

if (i != n - 1)

cout << " -> ";

}

cout << endl;

}

else

{

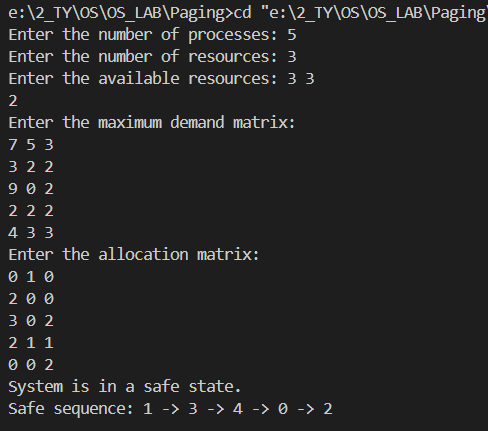
cout << "System is not in a safe state.\n";

}

return 0;

}

**Output:**

****