CSB 302: Operating System

Lab7: Banker's Algorithm for Deadlock Avoidance

Submitted By:

Name: **UMANG KUMAR**

Roll No: **201210051**

Branch: **CSE**

Semester: **5th Sem**

Submitted To: ***Dr. Rishav Singh***

Release Date: 03/11/2022 **Submitted Date:** 06/11/2022

NATIONAL INSTITUTE OF TECHNOLOGY DELHI

Department of Computer Science and Engineering

2022

Q1. Write a program in C to implement Banker's algorithm for deadlock avoidance.

Code:

```
#include <bits/stdc++.h>
using namespace std;

void solve() {
    int n;
    cin >> n;
    int m;
    cin >> m;

    // total available
    vector<int> availableRes(n);
    for (int i = 0; i < m; ++i) cin >> availableRes[i];

    vector<vector<int>> allocated(n, vector<int>(m, 0)),
        max(n, vector<int>(m, 0)), req(n, vector<int>(m, 0));
    for (auto& it : allocated) {
        for (auto& itt : it) {
            cin >> itt;
        }
    }

    for (auto& it : max) {
        for (auto& itt : it) {
            cin >> itt;
        }
    }

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            req[i][j] = max[i][j] - allocated[i][j];
        }
    }

    int processCompleted = 0;
```

```
    unordered_map<int, int> processCompletedMap;
    while (processCompleted < n) {
        int pCompletedInIteration = 0;
        for (int i = 0; i < n; ++i) {
            if (processCompletedMap.find(i) != processCompletedMap.end())
                continue;
            int resAvb = 0;
            for (int j = 0; j < m; ++j) {
                if (availableRes[j] >= req[i][j]) resAvb++;
            }
            if (resAvb == m) {
                processCompletedMap[i]++;
                for (int j = 0; j < m; ++j) {
                    availableRes[j] += allocated[i][j];
                }
                cout << "Process completed"
                     << " P" << i << endl;
                processCompleted++;
            }
        }
    }
}

int32_t main() {

    int tc = 1;
    while (tc--) {
        solve();
    }
    return 0;
}
```

Output:

```
Process completed P1
Process completed P3
Process completed P4
```

```
Process completed P0
Process completed P2
```