**Institute of Engineering & Management**
**Department of Computer Science & Engineering**
**Operating System Lab for 3rd year 6th semester 2019**
**Code: CS 693**

**Date:** 03/04/19

**WEEK-7**

**Assignment-1**

**Problem Statement:** Implement Banker's Algorithm

**Source Code:**

```
#include <iostream>
#include <vector>

bool find_safe(std::vector<int> avail, std::vector<std::vector<int>> &need,
           std::vector<std::vector<int>> &alloc, std::vector<int> &safe)
{
    int n = avail.size(), m = safe.size();
    std::vector<int> done(m,0);
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<m;j++)
        {
            if(done[j] == 1)
                continue;
            int flag = 0;
            for(int k=0;k<n;k++)
            {
                if(avail[k] < need[j][k])
                {
                    flag = 1;
                    break;
                }
            }
            if(flag == 0)
            {
                done[j] = 1;
                safe[i] = j;
                break;
            }
        }
        if(safe[i] == -1)
            return false;
        else{
            for(int j=0;j<n;j++)
                avail[j] += alloc[safe[i]][j];
        }
    }
    return true;
}

int main()
{
    std::cout<<"\n\t----Banker's Algorithm----\n";
    int n, m, allc=0, req;
    std::cout<<"Enter no. of resources types: ";
    std::cin>>n;
    std::cout<<"Enter no. of processes: ";
```

```
    std::cin>>m;
    std::vector<int> r(n,0), avail(n,0), safe(m,-1), req_v(n,0);
    std::vector<std::vector<int>> max(m,std::vector<int>(n,0)),
           alloc(m,std::vector<int>(n,0)), need(m,std::vector<int>(n,0));
    std::cout<<"Enter the total no of instances of each resource type: ";
    for(int i=0;i<n;i++)
        std::cin>>r[i];
    std::cout<<"Enter the max need of each process: \n";
    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            std::cin>>max[i][j];
    std::cout<<"Enter the allocated no. of resources: \n";
    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            std::cin>>alloc[i][j];
    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            need[i][j] = max[i][j]-alloc[i][j];
    std::cout<<"Need Matrix:\n";
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
            std::cout<<need[i][j]<<"\t";
        std::cout<<"\n";
    }
    for(int j=0;j<n;j++)
    {
        allc = 0;
        for(int i=0;i<m;i++)
            allc += alloc[i][j];
        avail[j] = r[j] - allc;
    }
    std::cout<<"available: ";
    for(auto i: avail)
        std::cout<<i<<" ";
    std::cout<<"\n";
    if(find_safe(avail, need, alloc, safe))
    {
        std::cout<<"\nSafe State!\nSafe Sequence:";
        for(int i=0;i<m;i++)
            std::cout<<" p"<<safe[i];
        std::cout<<"\n";
    }
    else
        std::cout<<"\nNot safe!\n";
    std::cout<<"\nEnter the requesting process no.: ";
    std::cin>>req;
    std::cout<<"Enter the request vector: ";
    for(int i=0;i<n;i++)
        std::cin>>req_v[i];
    for(int i=0;i<n;i++)
        alloc[req][i] += req_v[i];
    for(int i=0;i<n;i++)
        need[req][i] -= req_v[i];
    for(int i=0;i<n;i++)
        avail[i] -= req_v[i];
    if(find_safe(avail, need, alloc, safe))
        std::cout<<"Request is Granted!\n\n";
    else
        std::cout<<"Request is Denied!\n\n";
}
```

**Screen-Shot:**

```
        ----Banker's Algorithm----
Enter no. of resources types: 3
Enter no. of processes: 5
Enter the total no of instances of each resource type: 15 8 8
Enter the max need of each process:
5 6 3
8 5 6
4 8 2
7 4 3
4 3 3
Enter the allocated no. of resources:
2 1 0
3 2 3
3 0 2
3 2 0
1 0 1
Need Matrix:
3       5       3
5       3       3
1       8       0
4       2       3
3       3       2
available: 3 3 2

Safe State!
Safe Sequence: p4 p3 p0 p1 p2

Enter the requesting process no.: 4
Enter the request vector: 2 0 2
Request is Granted!
```