```
//Gaurav Rai 1706019
/* Question : (Resouce Request Algorithm)
Consider a system with five processes P0-P4 and 3 resources of type R1,R2 and R3.
Resource type R1 has 10 instances,R2 has 5 and R3 has 7 instances;
Suppose at time t0 following snapshot of the system has been taken
Process  Allocation     Max          Available
 P0         0 1 0       7 5 3          3 3 2
 P1         2 0 0       3 2 2
 P2         3 0 2       9 0 2
 P3         2 1 1       2 2 2
 P4         0 0 2       4 3 3
// Is the system in a safe state ? if yes, then find safe sequence...
// What will happen if process P1 request 1 additional instances of resources R1,0
of R2 and 2 of type R3 ??*/
#include<bits/stdc++.h>
using namespace std;
bool lessthan(int Need[],int Available[],int m){
    for(int i=0;i<m;i++){
        if(Need[i]>Available[i])
        return false;
    }
    return true;
}
int main(){
    int n=0,m=0;
    cout<<"Enter the number of Processes :";cin>>n;
    cout<<"Enter the number of Resources : ";cin>>m;
    int Available[m+1],AvailableC[m+1];
    cout<<"\n\tAvailable Number of Resources of : \n";
    for(int i=0;i<m;i++)
        {cout<<"R"<<i+1<<" : ";cin>>Available[i];AvailableC[i]=Available[i];}
    cout<<" Allocation Matrix\n";
    int Allocation[n+1][m+1],AllocationC[n+1][m+1];
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++)
            {cin>>Allocation[i][j];AllocationC[i][j]=Allocation[i][j];}
    cout<<" Max Req. Matrix\n";
    int Max[n+1][m+1],MaxC[n+1][m+1];
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++)
            {cin>>Max[i][j];MaxC[i][j]=Max[i][j];}
    cout<<"\tNeed Matrix\n";
    int Need[n+1][m+1],NeedC[n+1][m+1];
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++)
        { Need[i][j]=Max[i][j]-Allocation[i][j];
NeedC[i][j]=Need[i][j];cout<<Need[i][j]<<" ";}
            cout<<endl;
    }
    bool vis[n+1];int epid;
```

```
    memset(vis,false,sizeof(vis));
    vector<int> safeSeq;
    bool flag1=true;
    while(safeSeq.size()<n && flag1 ){
        flag1=false;
        for(int i=0;i<n ;i++){
            if(vis[i]==false && lessthan(Need[i],Available,m)){
                flag1=true;
                vis[i]=true;
                safeSeq.push_back(i);
                for(int j=0;j<m;j++)
                Available[j]+=Allocation[i][j];
            }
        }
    }
    if(safeSeq.size()!=n){
        cout<<"Deadlock !!! \n No SafeSequence Exists\n";
        cout<<"No extra Resource Can be provided ";
        cout<<"As System already in Deadlock\n";
    }
    else{
        cout<<"Safe Sequence :";
        for(int i=0;i<n ;i++){
            cout<<"->"<<"P"<<safeSeq[i];}
        cout<<"\nEnter the Process for which extra Resources are Required :
";cin>>epid;
        int required[m+1];
        cout<<"\n\tRequired Number of Resources of : \n";
        for(int i=0;i<m;i++){
            cout<<"R"<<i+1<<" : ";cin>>required[i];}
        if(lessthan(required,NeedC[epid],m)){
            memset(vis,false,sizeof(vis));
            safeSeq.clear();
            bool flag1=true;
            bool flag3=true;
            bool flag2=false;
            while(safeSeq.size()<n && flag1 ){
                flag1=false;
                for(int i=0;i<n ;i++){
                    if(lessthan(required,AvailableC,m) && flag3){
                        flag3=false;
                        flag1=true;
                        for(int i=0;i<m;i++)
                        {AvailableC[i]-=required[i];
                        AllocationC[epid][i]+=required[i];
                        NeedC[epid][i]-=required[i];}
                        flag2=true;
                    }
                    if(vis[i]==false && lessthan(NeedC[i],AvailableC,m) &&
(flag2==true || i!=epid)){
```

```
                    flag1=true;
                    vis[i]=true;
                    safeSeq.push_back(i);
                    for(int j=0;j<m;j++)
                    AvailableC[j]+=AllocationC[i][j];
                }
            }
        }
        cout<<"\nRequired : \n";
        for(int i=0;i<m ;i++)
        cout<<required[i]<<" ";
        cout<<"\n Allocation Matrix\n";
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++)
                cout<<AllocationC[i][j]<<" ";
            cout<<endl;
        }
        cout<<" Max Req. Matrix\n";
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++)
                cout<<MaxC[i][j]<<" ";
            cout<<endl;
        }
        cout<<"\tNeed Matrix\n";
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++)
               cout<<NeedC[i][j]<<" ";
            cout<<endl;
        }
        if(safeSeq.size()!=n){
            cout<<"Deadlock !!! \n No SafeSequence Exists\n";
            cout<<"No extra Resource Can be provided otherwise OS would go in
Deadlock\n";
        }
        else{
            cout<<"Yes! Immediate resource Request for "<<epid<<" can be
completed \n";
            cout<<"Safe Sequence :";
            for(int i=0;i<n ;i++)
            {cout<<"->"<<"P"<<safeSeq[i];}
        }
    }
    else
        cout<<"Resource Request Cant be fullfilled\n";
}
    return 0;
}
```