N Vijay Chowdary

Implement deadlock avoidance, detection algorithms using System V IPC facilities.

Code:

#include<bits/stdc++.h>

#include<iostream>

#include<vector>

using namespace std;

int n,m;

vector<vector<int>>alloc,Max,need;

vector<int> avail;

bool check(vector<int> &a,vector<int> &b){

  for(int i = 0;i < m;i++){

    if(a[i]<b[i]) return false;

  }

  return true;

}

void add(vector<int> &a,vector<int> &b){

  for(int i = 0;i < m;i++) a[i] += b[i];

}

void Banker(){

  for(int i = 0;i < n;i++){

    vector<int> temp;

    for(int j = 0;j < m;j++)

      temp.push\_back(Max[i][j] - alloc[i][j]);

    need.push\_back(temp);

  }

  int completed = 0;

  int flag[n];

  for(int i = 0;i < n;i++) flag[i]=0;

  bool issafe = true;

  vector<int>safeseq;

  while(completed != n){

    bool tag = false;

    for(int i = 0;i < n;i++){

      if(flag[i]) continue;

      if(check(avail,need[i])){

        safeseq.push\_back(i);

        add(avail,alloc[i]);

        tag = true;

        flag[i] = 1;

        completed++;

      }

    }

    if(!tag){

      issafe = false;

      break;

    }

  }

  if(!issafe){

    cout<<"Deadlock!!!\n";

    return;

  }

  cout<<"Safe \n";

  cout<<"Sequence: ";

  for(int i = 0;i < n;i++) cout<<safeseq[i]<<" ";

  cout<<"\n";

}

int main(){

  cout<<"Number of processes and resources: ";

  cin>>n>>m;

  cout<<"Allocation Matrix: \n";

  int input;

  for(int i = 0;i < n;i++){

    vector<int>temp;

    for(int j = 0;j < m;j++){

      cin>>input;

      temp.push\_back(input);

    }

    alloc.push\_back(temp);

  }

  cout<<"Max claim matrix: \n";

  for(int i = 0;i < n;i++){

    vector<int> temp;

    for(int j = 0;j < m;j++){

      cin>>input;

      temp.push\_back(input);

    }

    Max.push\_back(temp);

  }

  cout<<"enter availability: \n";

  for(int i = 0;i < m;i++){

    cin>>input;

    avail.push\_back(input);

  }

  Banker();

}

Output:

