###### Experiment Number: 05

###### TITLE: Bankers Algorithm

# Code

*/\**

*Initial Resource Instance availability :*

*A   B   C*

*10  5   7*

*========================================================*

*Allocation(A)       Max(M)       Need = M - A*

*========================================================*

*Process     A   B   C       A   B   C       A   B   C*

*P0          0   1   0       7   5   3       7   4   3*

*P1          2   0   0       3   2   2       1   2   2*

*P2          3   0   2       9   0   2       6   0   0*

*P3          2   1   1       2   2   2       0   1   1*

*P4          0   0   2       4   3   3       4   3   1*

*--------------------------------------------------------*

*Total       7   2   5*

*========================================================*

*Instance availability after resource allocation :*

*A            B           C*

*10 - 7 = 3   5 - 2 = 3   7 - 5 = 2*

*\*/*

*#include* <iomanip>

*#include* <iostream>

using namespace std;

int main()

{

    int i, j, k;

    const int n = 5; *// number of processes*

    const int m = 3; *// number of resources*

    int allocation[n][m] = {{0, 1, 0}, *// Allocation Matrix*

                            {2, 0, 0},

                            {3, 0, 2},

                            {2, 1, 1},

                            {0, 0, 2}}; *// Each row represents a process*

    int max[n][m] = {{7, 5, 3}, *// MAX Matrix*

                     {3, 2, 2},

                     {9, 0, 2},

                     {2, 2, 2},

                     {4, 3, 3}}; *// Maximum resources that can be allocated*

    int available[m] = {3, 3, 2}; *// Available resources at start*

    int finish[n] = {0};

    int ans[n] = {0};

    int idx = 0;

    int need[n][m]; *// Calculating Need matrix*

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

    {

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

        {

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

        }

    }

    int y = 0;

*for* (int k = 0; k < 5; k++)

    {

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

        {

*if* (finish[i] == 0)

            {

                int flag = 0;

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

                {

*if* (need[i][j] > available[j])

                    {

                        flag = 1; *// if needed resources are more in number than the available ones, move to the next process*

*break*;

                    }

                }

*if* (flag == 0)

                { *// if available resources fulfilled the need*

                    ans[idx++] = i; *// the index of process, that has been allocated the resources*

*for* (int y = 0; y < m; y++)

                    {

                        available[y] += allocation[i][y];

                    }

                    finish[i] = 1;

                }

            }

        }

    }

    bool flag = true;

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

    {

*if* (finish[i] == 0)

        {

            flag = false;

            cout << "System is in deadlock !!";

*break*;

        }

    }

*if* (flag == true)

    {

        cout << "System is in safe state and following is the safe sequence: ";

*for* (int i = 0; i < n - 1; i++)

        {

            cout << "P" << ans[i] << " -> ";

        }

        cout << "P" << ans[n - 1] << endl;

    }

*return* 0;

}

# Output

