****

**NAME: -** Darji Akshatkumar Hiteshbhai

**RollNo: -** 23MCD001

**Branch: -** M.tech-CSE**(Data Science)**

**Subject: -** Complexity Theory & Algorithms

**Practical-9**

**Aim:** Given two sequences X and Y, find the longest common subsequence (LCS) of X and Y using dynamic programming.

**Code for LCS –**

//

// Created by AKSHAT on 11/4/2023.

//

#include<bits/stdc++.h>

using namespace std;

void PRINT\_LCS(vector<vector<char>>& b, const string& X, int i, int j) {

    if (i == 0 || j == 0) {

        return;

    }

    if (b[i][j] == 'D') {

        PRINT\_LCS(b, X, i - 1, j - 1);

        cout << X[i - 1];

    } else if (b[i][j] == 'U') {

        PRINT\_LCS(b, X, i - 1, j);

    } else {

        PRINT\_LCS(b, X, i, j - 1);

    }

}

void PRINT\_DP\_TABLE(vector<vector<int>>& c, vector<vector<char>>& b, const string& X, const string& Y) {

    int m = X.length();

    int n = Y.length();

    cout << "       ";

    for (int j = 0; j < n; j++) {

        cout << Y[j] << "   ";

    }

    cout << endl;

    for (int i = 0; i <= m; i++) {

        if (i == 0) {

            cout << "  ";

        } else {

            cout << X[i-1] << " ";

        }

        for (int j = 0; j <= n; j++) {

            cout << c[i][j] << b[i][j] << "  ";

        }

        cout << endl;

    }

}

int LCS(const string& X, const string& Y) {

    int m = X.length();

    int n = Y.length();

    vector<vector<int>> c(m + 1, vector<int>(n + 1, 0)); //for value

    vector<vector<char>> b(m + 1, vector<char>(n + 1, ' ')); //for sign

    for (int i = 1; i <= m; i++) {

        for (int j = 1; j <= n; j++) {

            if (X[i - 1] == Y[j - 1]) {

                c[i][j] = c[i - 1][j - 1] + 1;

                b[i][j] = 'D'; // Diagonal

            } else if (c[i - 1][j] >= c[i][j - 1]) {

                c[i][j] = c[i - 1][j];

                b[i][j] = 'U'; // Up

            } else {

                c[i][j] = c[i][j - 1];

                b[i][j] = 'L'; // Left

            }

        }

    }

    PRINT\_DP\_TABLE(c, b, X, Y);

    cout << "Longest Common Subsequence: ";

    PRINT\_LCS(b, X, m, n);

    cout << endl;

    // Return the length of the LCS

    return c[m][n];

}

int main() {

    string X = "PQRSPQQR";

    string Y = "RSPQRS";

    int length = LCS(X, Y);

    cout << "Length of LCS: " << length << endl;

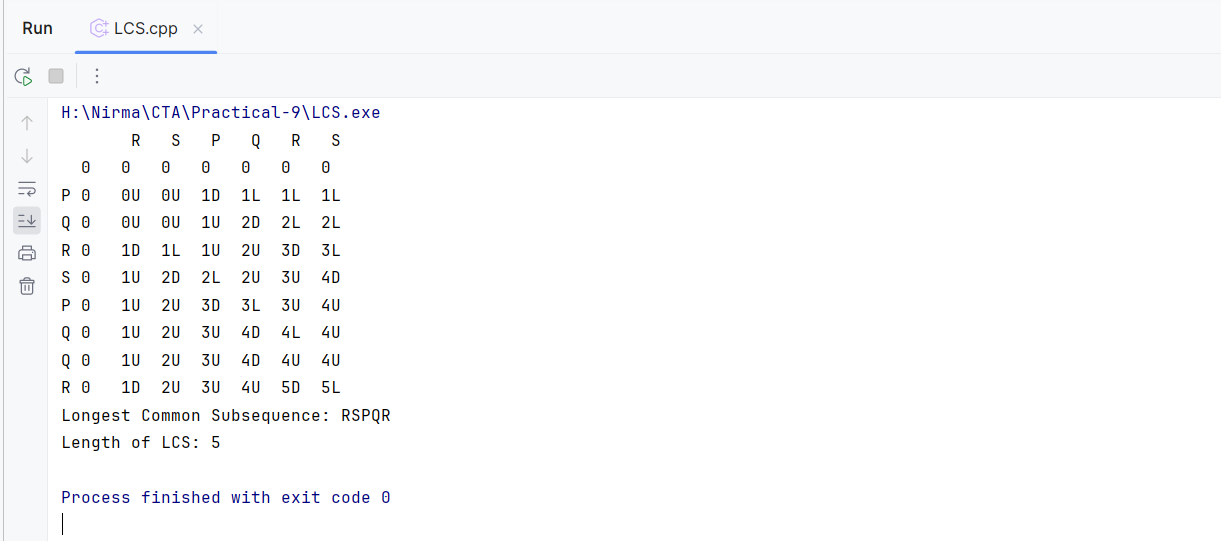
    return 0;

}

**Output –**

**Test Case – 1**

For x = PQRSPQQR and y = RSPQRS

****