DAA Lab

Practical 4

Amber Shukla

A1-B3-42

**Aim:** Implement Longest Common Subsequence (LCS) algorithm to find the length and LCS for DNA sequences.

**Code:**

#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

vector<vector<vector<int>>> arr;

string text1, text2;

void backTrack(int i, int j)

{

    if (i == 0 || j == 0)

        return;

    int temp = arr[i][j][1];

    if (temp == 0)

        backTrack(i - 1, j - 1),

            cout << text1[i-1];

    else if (temp == 1)

        backTrack(i, j - 1);

    else

        backTrack(i - 1, j);

}

int main()

{

    cout << "Enter String 1: ";

    cin >> text1;

    cout << "Enter String 2: ";

    cin >> text2;

    int size1 = text1.size();

    int size2 = text2.size();

    arr.resize(size1 + 1, vector<vector<int>>(size2 + 1, vector<int>(2, 0)));

    // 0 Means Diagonal , 1 Means Upper , 2 Means Side

    for (int i = 1; i <= size1; i++)

    {

        for (int j = 1; j <= size2; j++)

        {

            if (text1[i - 1] == text2[j - 1]) // Diagonal + 1

            {

                arr[i][j][0] = arr[i - 1][j - 1][0] + 1;

                arr[i][j][1] = 0;

            }

            else

            {

                if (arr[i - 1][j][0] >= arr[i][j - 1][0]) // Up is Bigger

                {

                    arr[i][j][0] = arr[i - 1][j][0];

                    arr[i][j][1] = 2;

                }

                else // Left is Bigger

                {

                    arr[i][j][0] = arr[i][j - 1][0];

                    arr[i][j][1] = 1;

                }

            }

        }

    }

    cout << "Length of LCS is: " << arr[size1][size2][0] << endl;

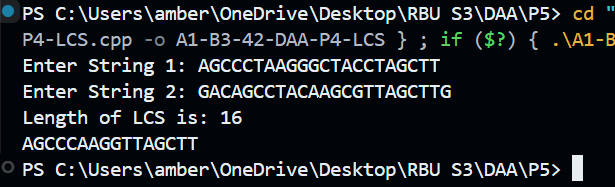
    // Print LCS

    backTrack(size1, size2);

    return 0;

}

**Output:**



**PART 2**

**Aim:** Find the longest repeating subsequence (LRS). Consider it as a variation of the

longest common subsequence (LCS) problem.

**Code:** (Just condition added to check if **i != j)**

#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

vector<vector<vector<int>>> arr;

string text1;

void backTrack(int i, int j)

{

    if (i == 0 || j == 0)

        return;

    int temp = arr[i][j][1];

    if (temp == 0)

        backTrack(i - 1, j - 1),

            cout << text1[i-1];

    else if (temp == 1)

        backTrack(i, j - 1);

    else

        backTrack(i - 1, j);

}

int main()

{

    cout << "Enter String: ";

    cin >> text1;

    int size1 = text1.size();

    arr.resize(size1 + 1, vector<vector<int>>(size1 + 1, vector<int>(2, 0)));

    // 0 Means Diagonal , 1 Means Upper , 2 Means Side

    for (int i = 1; i <= size1; i++)

    {

        for (int j = 1; j <= size1; j++)

        {

            if (text1[i - 1] == text1[j - 1] && i != j) // Diagonal + 1

            {

                arr[i][j][0] = arr[i - 1][j - 1][0] + 1;

                arr[i][j][1] = 0;

            }

            else

            {

                if (arr[i - 1][j][0] >= arr[i][j - 1][0]) // Up is Bigger

                {

                    arr[i][j][0] = arr[i - 1][j][0];

                    arr[i][j][1] = 2;

                }

                else // Left is Bigger

                {

                    arr[i][j][0] = arr[i][j - 1][0];

                    arr[i][j][1] = 1;

                }

            }

        }

    }

    cout << "Length of LCS is: " << arr[size1][size1][0] << endl;

    // Print LCS

    backTrack(size1, size1);

    return 0;

}

**Output:**

