/\* Dynamic Programming implementation of LCS problem \*/

#include<iostream>

#include<cstring>

#include<cstdlib>

using namespace std;

/\* Returns length of LCS for X[0..m-1], Y[0..n-1] \*/

void lcs( char \*X, char \*Y, int m, int n )

{

   int L[m+1][n+1];

   /\* Following steps build L[m+1][n+1] in bottom up fashion. Note

      that L[i][j] contains length of LCS of X[0..i-1] and Y[0..j-1] \*/

   for (int i=0; i<=m; i++)

   {

     for (int j=0; j<=n; j++)

     {

       if (i == 0 || j == 0)

         L[i][j] = 0;

       else if (X[i-1] == Y[j-1])

         L[i][j] = L[i-1][j-1] + 1;

       else

         L[i][j] = max(L[i-1][j], L[i][j-1]);

     }

   }

   // Following code is used to print LCS

   int index = L[m][n];

   // Create a character array to store the lcs string

   char lcs[index+1];

   lcs[index] = ''; // Set the terminating character

   // Start from the right-most-bottom-most corner and

   // one by one store characters in lcs[]

   int i = m, j = n;

   while (i > 0 && j > 0)

   {

      // If current character in X[] and Y are same, then

      // current character is part of LCS

      if (X[i-1] == Y[j-1])

      {

          lcs[index-1] = X[i-1]; // Put current character in result

          i--; j--; index--;     // reduce values of i, j and index

      }

      // If not same, then find the larger of two and

      // go in the direction of larger value

      else if (L[i-1][j] > L[i][j-1])

         i--;

      else

         j--;

   }

   // Print the lcs

   cout << "LCS of " << X << " and " << Y << " is " << lcs;

}