Name	MAAZ SHAIKH
UID no.	2021700059
Experiment No.	04

AIM:	To find Longest common subsequence using dynamic Programming.
PROBLEM STATEMENT :	For the given two strings, find the longest common subsequence length and also print the same.
ALGORITHM/ THEORY:	The longest common subsequence (LCS) is defined as the longest subsequence that is common to all the given sequences, provided that the elements of the subsequence are not required to occupy consecutive positions within the original sequences. If `S1` and `S2` are the two given sequences then, `Z` is the common subsequence of `S1` and `S2` if `Z` is a subsequence of both `S1` and `S2`. Furthermore, `Z` must be a **strictly increasing sequence** of the indices of both `S1` and `S2`. In a strictly increasing sequence, the indices of the elements chosen from the original sequences must be in ascending order in `Z`. If `S1 = {B, C, D, A, A, C, D}` Then, `{A, D, B}` cannot be a subsequence of `S1` as the order of the elements is not the same (ie. not strictly increasing sequence).

	Let us understand LCS with an example. If S1 = {B, C, D, A, A, C, D} S2 = {A, C, D, B, A, C} Then, common subsequences are {B, C}, {C, D, A, C}, {D, A, C}, {A, A, C}, {A, C}, {C, D}, ... Among these subsequences, {C, D, A, C} is the longest common subsequence. We are going to find this longest common subsequence using dynamic programming.

PROGRAM:	`#include <stdio.h>`
	`#include <string.h>`
	
	`#define MAX_LEN 1000`
	
	`int max(int a, int b) {`
	`  return (a > b) ? a : b;`
	`}`
	
	`// function to find the LCS of two strings`
	`int LCS(char arr1[], char arr2[], int m, int n, char lcs_string[]) {`
	`  int lcs[m+1][n+1];`
	`  int i, j;`
	
	`  // building the lcs array`

```
    for (i = 0; i <= m; i++) {
        for (j = 0; j <= n; j++) {
            if (i == 0 || j == 0)
                lcs[i][j] = 0;
            else if (arr1[i-1] == arr2[j-1]) {
                lcs[i][j] = lcs[i-1][j-1] + 1;
                lcs_string[lcs[i][j]-1] = arr1[i-1];  // adding
the character to LCS string
            }
            else
                lcs[i][j] = max(lcs[i-1][j], lcs[i][j-1]);
        }
    }

    // adding null terminator to the LCS string
    lcs_string[lcs[m][n]] = '\0';

    // returning the LCS length
    return lcs[m][n];
}

int main() {
    char str1[MAX_LEN], str2[MAX_LEN], lcs_string[MAX_LEN];
    int m, n, lcs_length;

    printf("Enter the first string: ");
    fgets(str1, MAX_LEN, stdin);
    m = strlen(str1) - 1;  // -1 to remove the newline
character

    printf("Enter the second string: ");
    fgets(str2, MAX_LEN, stdin);
    n = strlen(str2) - 1;

    lcs_length = LCS(str1, str2, m, n, lcs_string);

    printf("The length of the LCS is: %d\n", lcs_length);
    printf("The LCS string is: %s\n", lcs_string);

    return 0;
}
```

RESULT:

```
PS C:\Users\maazs\OneDrive\Desktop\Studies\DAA\DAA Coding> cd "c:\Users\maazs\OneDrive\Desktop\Studies\DAA\DAA Coding\" ; if ($?) { gcc lcs.c
-o lcs } ; if ($?) { .\lcs }
Enter the first string: abababaa
Enter the second string: babbab
The length of the LCS is: 5
The LCS string is: babba
```

CONCLUSION:	I learned how to find Longest Common subsequence between two strings using dynamic Programming.