Practical no: 5

Name: Saloni Vishwakarma
Batch-Roll no: C1-13
Subject: DAA Lab

Aim: To implement Longest Common Subsequence algorithm using recursion in C. Display the lcs table, string and its length.

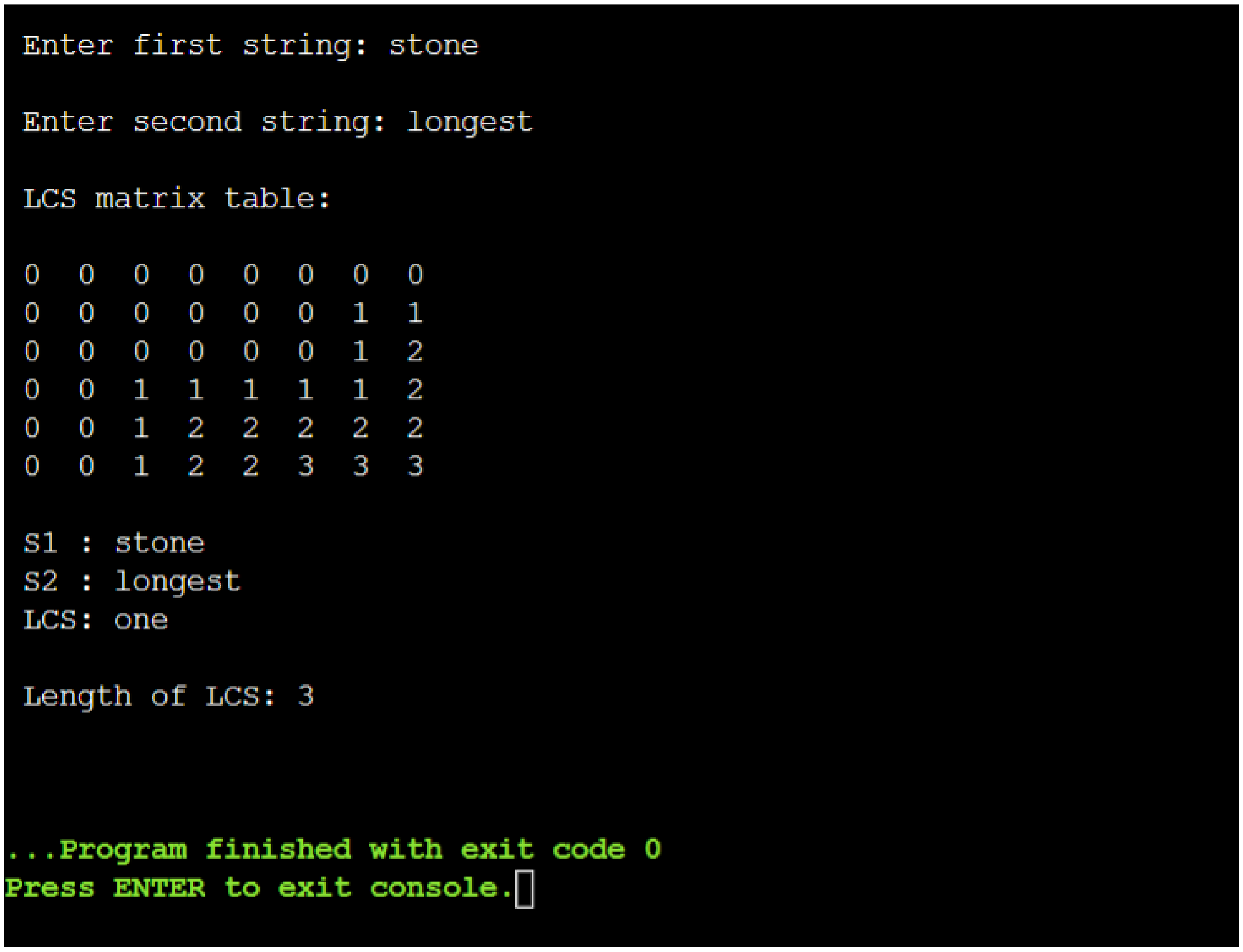
Code and Output:

```
#include <stdio.h>
#include <string.h>
#define MAX_ROWS 100
#define MAX_COLS 100

void printMatrix(int matrix[MAX_ROWS][MAX_COLS], int rows, int cols)
{
   int i,j;
   for ( i = 0; i < rows; i++) {
      for ( j = 0; j < cols; j++) {
         printf(" %d ", matrix[i][j]);
      }
      printf("\n");
   }
}
void longestCommonSubsequence(char *S1, char *S2)
{
   int len1 = strlen(S1);
   int len2 = strlen(S2);
   int matrix[MAX_ROWS][MAX_COLS];
   int i,j;
   for ( i = 0; i <= len1; i++) {
      for (j = 0; j <= len2; j++) {
         if (i == 0 || j == 0)
            matrix[i][j] = 0;
         else if (S1[i - 1] == S2[j - 1])
            matrix[i][j] = matrix[i - 1][j - 1] + 1;
            else
                matrix[i][j] = (matrix[i - 1][j] > matrix[i][j - 1]) ? matrix[i - 1][j] : matrix[i][j -1];
        }
    }
    printf("\n LCS matrix table:\n\n");
    printMatrix(matrix, len1 + 1, len2 + 1);
}
    int i, j, m, n, LCS_table[20][20];
    /*char S1[20] = "abaaba", S2[20] = "babbab", */char b[20][20];
void lcsAlgo(char *S1, char *S2)
{
    m = strlen(S1);
    n = strlen(S2);
    for (i = 0; i <= m; i++)
        LCS_table[i][0] = 0;
    for (i = 0; i <= n; i++)
        LCS_table[0][i] = 0;
    for (i = 1; i <= m; i++)
        for (j = 1; j <= n; j++) {
            if (S1[i - 1] == S2[j - 1])
            {
                LCS_table[i][j] = LCS_table[i - 1][j - 1] + 1;
            }
            else if (LCS_table[i - 1][j] >= LCS_table[i][j - 1])
            {
                LCS_table[i][j] = LCS_table[i - 1][j];
            }
            else
            {
                LCS_table[i][j] = LCS_table[i][j - 1];
            }
        }
    int index = LCS_table[m][n];
    char lcsAlgo[index + 1];
    lcsAlgo[index] = '\0';
    int i = m, j = n;
    while (i > 0 && j > 0)
    {
        if (S1[i - 1] == S2[j - 1]) {
        lcsAlgo[index - 1] = S1[i - 1];
        i--;
        j--;
        index--;
    }
    else if (LCS_table[i - 1][j] > LCS_table[i][j - 1])
        i--;
    else
        j--;
}

// Printing the sub sequences
printf("\n S1 : %s \n S2 : %s \n", S1, S2);
printf(" LCS: %s \n", lcsAlgo);
}
int max(int a, int b) {
return (a>b) ?a:b;
}

int lcs(char* X, char* Y, int m, int n)
{
    if (m== 0|| n== 0)
        return 0;
    if (X[m - 1] == Y[n - 1])
        return 1 + lcs(X, Y, m - 1, n - 1);
    else
        return max(lcs(X, Y, m, n - 1),
    lcs(X, Y, m - 1, n));
}
int main()
{
    char S1[20],S2[20];
    printf("\n Enter first string: ");
    scanf("%s",S1);
    printf("\n Enter second string: ");
    scanf("%s",S2);
    int m=strlen(S1);
    int n = strlen(S2);
    longestCommonSubsequence(S1, S2);
    int length = lcs(S1, S2, m, n);
    lcsAlgo(S1,S2);
    printf("\n Length of LCS: %d\n", length);
    printf("\n");
    return 0;
}
```



```
Enter first string: stone

Enter second string: longest

LCS matrix table:

0   0   0   0   0   0   0   0
0   0   0   0   0   0   1   1
0   0   0   0   0   0   1   2
0   0   1   1   1   1   1   2
0   0   1   2   2   2   2   2
0   0   1   2   2   3   3   3

S1 : stone
S2 : longest
LCS: one

Length of LCS: 3



...Program finished with exit code 0
Press ENTER to exit console.
```