Name: Sriram Gopalkrishnan Iyer
Section: A3
Batch: B3
Roll No: 38

Ramdeobaba University, Nagpur
Department of Computer Science and Engineering
Session: 2024-2025

Design and Analysis of Algorithms Lab **III Semester**

PRACTICAL NO. 5

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

Problem Statement:

(i) DNA sequences can be viewed as strings of A, C, G, and T characters, which represent nucleotides. Finding the similarities between two DNA sequences are an important computation performed in bioinformatics.

[Note that a subsequence might not include consecutive elements of the original sequence.]

TASK 1: Find the similarity between the given X and Y sequence.

X=AGCCCTAAGGGCTACCTAGCTT

Y= GACAGCCTACAAGCGTTAGCTTG

Output: Cost matrix with all costs and direction, final cost of LCS and the LCS.

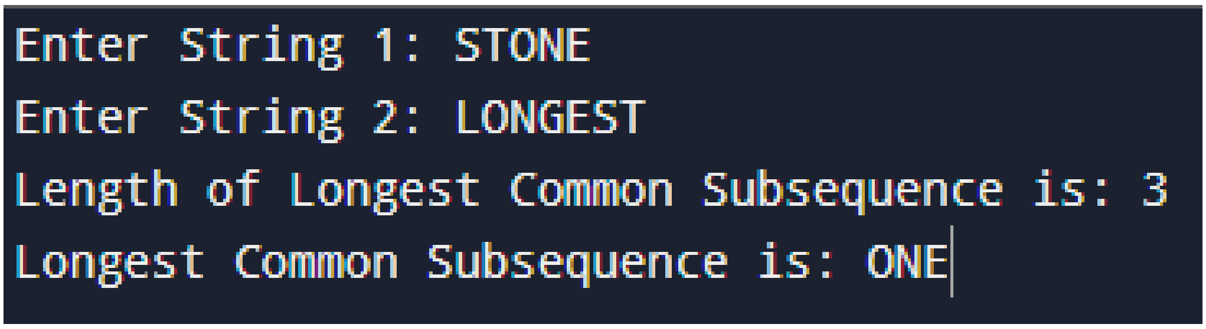
Length of LCS=16

Code:

```
#include <stdio.h>

#include <string.h>
```

```c
#define MAX 100

#define DIAG '\\'
#define UP '|'
#define LEFT '-'

char b[MAX][MAX];

int lcs_length(char str1[MAX], char str2[MAX]) {
    int m = strlen(str1);
    int n = strlen(str2);

    int c[MAX + 1][MAX + 1];

    for(int i = 1; i <= m; i++)
        c[i][0] = 0;
    for(int j = 0; j <= n; j++)
        c[0][j] = 0;

    for(int i = 1; i <= m; i++) {
        for(int j = 1; j <= n; j++) {
            if(str1[i - 1] == str2[j - 1]) {
                c[i][j] = c[i - 1][j - 1] + 1;
                b[i][j] = DIAG;
            } else if(c[i - 1][j] >= c[i][j - 1]) {
                c[i][j] = c[i - 1][j];
                b[i][j] = UP;
            } else {
                c[i][j] = c[i][j - 1];
                b[i][j] = LEFT;
            }
        }
    }

    return c[m][n];
}

void print_lcs(char b[MAX][MAX], char str1[MAX], int i, int j)
{
    if(i == 0 || j == 0)
        return;

    if(b[i][j] == DIAG) {
        print_lcs(b, str1, i - 1, j - 1);
        printf("%c", str1[i - 1]);
    } else if(b[i][j] == UP) {
        print_lcs(b, str1, i - 1, j);
    } else {
        print_lcs(b, str1, i, j - 1);
    }
}

int main() {
    char str1[MAX], str2[MAX];


    printf("Enter String 1: ");

    scanf("%s", str1);

    printf("Enter String 2: ");

    scanf("%s", str2);

    int length = lcs_length(str1, str2);


      printf("Length of Longest Common Subsequence is: %d\n",
length);

    printf("Longest Common Subsequence is: ");

    print_lcs(b, str1, strlen(str1), strlen(str2));


    return 0;

}
```

Output:



```
Enter String 1: STONE
Enter String 2: LONGEST
Length of Longest Common Subsequence is: 3
Longest Common Subsequence is: ONE
```

TASK-2: Find the longest repeating subsequence (LRS). Consider it as a variation of the longest common subsequence (LCS) problem.

Let the given string be S. You need to find the LRS within S. To use the LCS framework, you effectively compare S with itself. So, consider string1 = S and string2 = S.

Example:

AABCBDC

LRS= ABC or ABD

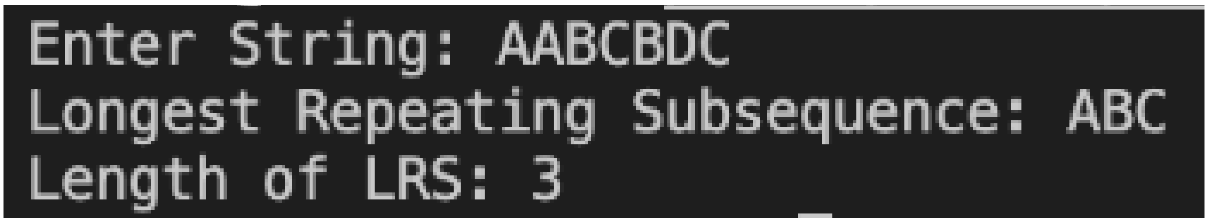
Code:

```
#include <stdio.h>

#include <string.h>


#define MAX 100


void printLRS(char str[], int dp[MAX][MAX], int n) {

   int i = n, j = n;

   char lrs[MAX];

   int index = 0;


   while (i > 0 && j > 0) {

       if (str[i-1] == str[j-1] && i != j) {

           lrs[index++] = str[i-1];

           i--; j--;

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

           i--;

       } else {

           j--;

       }

   }
```

```c
   for (int k = index-1; k >= 0; k--)
       printf("%c", lrs[k]);
   printf("\n");
}

int LRS(char str[]) {
   int n = strlen(str);
   int dp[MAX][MAX];

   for(int i = 0; i <= n; i++)
       for(int j = 0; j <= n; j++)
           dp[i][j] = 0;

   for(int i = 1; i <= n; i++) {
       for(int j = 1; j <= n; j++) {
           if(str[i-1] == str[j-1] && i != j) {
               dp[i][j] = 1 + dp[i-1][j-1];
           } else {
               if(dp[i-1][j] > dp[i][j-1]) {
                   dp[i][j] = dp[i-1][j];
               } else {
                   dp[i][j] = dp[i][j-1];
               }
           }
```

```c
        }
    }

    printf("Longest Repeating Subsequence: ");
    printLRS(str, dp, n);

    return dp[n][n];
}

int main() {
    char str[MAX];

    printf("Enter String: ");
    scanf("%s", str);

    int length = LRS(str);
    printf("Length of LRS: %d\n", length);

    return 0;
}
```

Output:

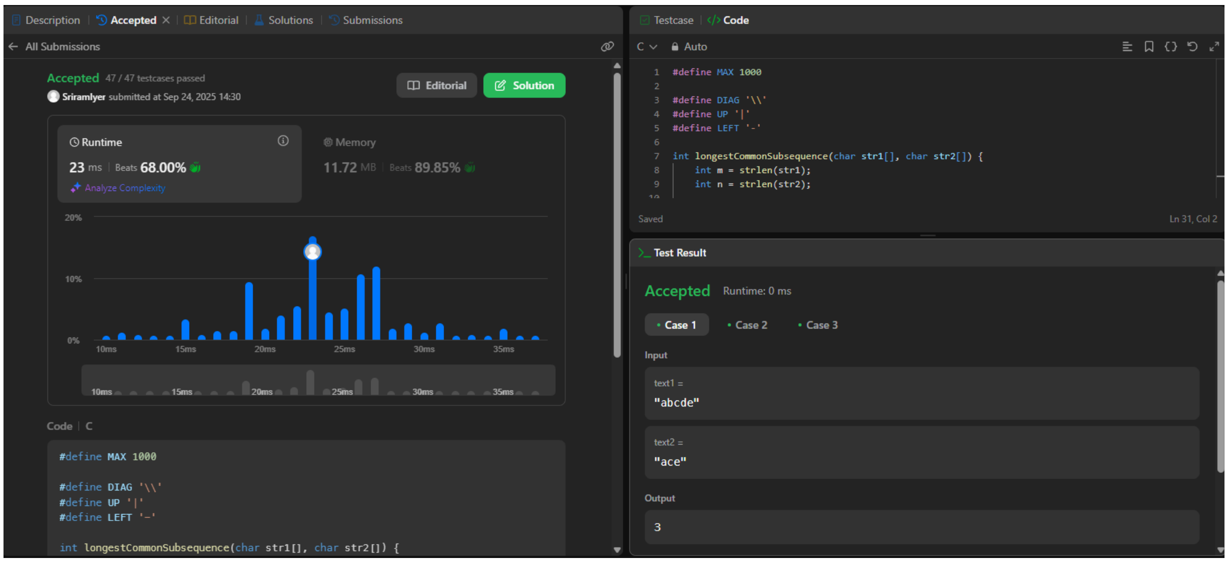


```
Enter String: AABCBDC
Longest Repeating Subsequence: ABC
Length of LRS: 3
```

LeetCode Assesment:

https://leetcode.com/problems/longest-common-subsequence/description/

Output:



Accepted 47 / 47 testcases passed

SriramIyer submitted at Sep 24, 2025 14:30

Runtime
23 ms | Beats 68.00%

Memory
11.72 MB | Beats 89.85%

Code | C

```
#define MAX 1000

#define DIAG '\\'
#define UP '|'
#define LEFT '-'

int longestCommonSubsequence(char str1[], char str2[]) {
    int m = strlen(str1);
    int n = strlen(str2);
```

Accepted Runtime: 0 ms

Input

text1 =
"abcde"

text2 =
"ace"

Output

3