

# DAA PRACTICAL 5

## TASK 1:

Find the similarity between the given X and Y sequence.

X=AGCCCTAAGGGCTACCTAGCTT

Y= GACAGCCTACAAGCGTTAGCTTG

```
//longest common sequence(LCS)

#include <stdio.h>
#include <string.h>

int max(int a, int b)
{
    return a > b ? a : b;
}

void LCS(char *X, char *Y) {
    int m = strlen(X), n = strlen(Y);
    int dp[m+1][n+1];

    // Fill DP table
    for (int i = 0; i <= m; i++)
        for (int j = 0; j <= n; j++)
            if (i == 0 || j == 0)
                dp[i][j] = 0;
            else if (X[i-1] == Y[j-1])
                dp[i][j] = dp[i-1][j-1] + 1;
            else
                dp[i][j] = max(dp[i-1][j], dp[i][j-1]);

    // Backtrack to get LCS
    int i = m, j = n, k = dp[m][n];
    char lcs[k+1];
    lcs[k] = '\0';

    while (i > 0 && j > 0) {
        if (X[i-1] == Y[j-1]) {
```

```

        lcs[--k] = X[i-1];
        i--; j--;
    } else if (dp[i-1][j] > dp[i][j-1])
        i--;
    else
        j--;
}

printf("LCS Length: %d\n", dp[m][n]);
printf("LCS: %s\n", lcs);
}

int main() {
    char X[] = "AGCCCTAAGGGCTACCTAGCTT";
    char Y[] = "GACAGCCTACAAGCGTTAGCTTG";
    LCS(X, Y);
    return 0;
}

```

```

[Running] cd "c:\Users\DT USER\Desktop\1A333333\DA
LCS Length: 16
LCS: GCCCTAAGCTTAGCTT

[Done] exited with code=0 in 0.87 seconds

```

## TASK-2:

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

```
// longest repeating sequence (LRS)

#include <stdio.h>
#include <string.h>

int max(int a, int b)
{
    return a > b ? a : b;
}

void LRS(char *str) {
    int n = strlen(str);
    int dp[n+1][n+1];
    // Fill DP table
    for (int i = 0; i <= n; i++)
        for (int j = 0; j <= n; j++)
            if (i == 0 || j == 0)
                dp[i][j] = 0;
            else if (str[i-1] == str[j-1] && i != j)
                dp[i][j] = dp[i-1][j-1] + 1;
            else
                dp[i][j] = max(dp[i-1][j], dp[i][j-1]);

    // Backtrack to get LRS
    int i = n, j = n, k = dp[n][n];
    char lrs[k+1];
    lrs[k] = '\0';
    while (i > 0 && j > 0) {
        if (str[i-1] == str[j-1] && i != j) {
            lrs[--k] = str[i-1];
            i--; j--;
        } else if (dp[i-1][j] > dp[i][j-1])
            i--;
        else
            j--;
    }

    printf("LRS Length: %d\n", dp[n][n]);
    printf("LRS: %s\n", lrs);
}
```

```
int main() {
    char S[] = "AABCBDC";
    LRS(S);
    return 0;
}
```

```
[Running] cd "c:\Users\DT USER\Desktop\1A333333\USER\Desktop\1A333333\DAA\A333333\daa 4\daa5\
LRS Length: 3
LRS: ABC

[Done] exited with code=0 in 0.241 seconds
```

## LEETCODE:

Problem List

1143. Longest Common Subsequence

Solved

Description

Editorial

Solutions

Submissions

14.6K

239

187 Online

Accepted

47 / 47 testcases passed

ansha\_lanjewar submitted at Sep 24, 2025 14:10

Editorial

Solution

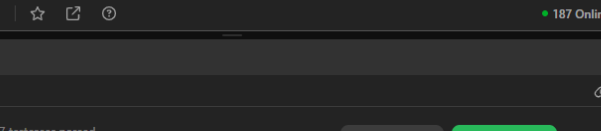
Runtime

23 ms | Beats 68.00%

Analyze Complexity

Memory

12.35 MB | Beats 33.54%



Code

C

Auto

```
1 int longestCommonSubsequence(char * text1, char * text2) {
2     int m = strlen(text1);
3     int n = strlen(text2);
4
5     int dp[m + 1][n + 1];
6
7     for (int i = 0; i <= m; i++)
8         for (int j = 0; j <= n; j++)
9             dp[i][j] = 0;
10
11     for (int i = 1; i <= m; i++) {
12         for (int j = 1; j <= n; j++) {
```

Saved

Ln 1, Col 1

Testcase

Test Result

Accepted

Runtime: 0 ms

Case 1

Case 2

Case 3

Input

text1 = "abcde"

text2 = "ace"

Output