**Aim** : Implement a dynamic algorithm for Longest Common Subsequence (LCS) to find the

length and LCS for DNA sequences.

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

**X =** AGCCCTAAGGGCTACCTAGCTT

**Y =** GACAGCCTACAAGCGTTAGCTTG

**Code :**

def LCS(X, Y):

    m, n = len(X), len(Y)

    C = [[0]\*(n+1) for \_ in range(m+1)]

    for i in range(1, m+1):

        for j in range(1, n+1):

            if X[i-1] == Y[j-1]:

                C[i][j] = C[i-1][j-1] + 1

            else:

                C[i][j] = max(C[i-1][j], C[i][j-1])

    i, j = m, n

    lcs = []

    while i > 0 and j > 0:

        if X[i-1] == Y[j-1]:

            lcs.append(X[i-1])

            i -= 1

            j -= 1

        elif C[i-1][j] >= C[i][j-1]:

            i -= 1

        else:

            j -= 1

    return C[m][n], ''.join(reversed(lcs))

X = "AGCCCTAAGGGCTACCTAGCTT"

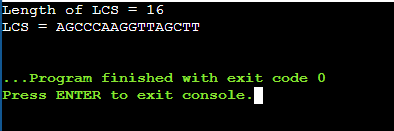
Y = "GACAGCCTACAAGCGTTAGCTTG"

length, lcs\_seq = LCS(X, Y)

print("Length of LCS =", length)

print("LCS =", lcs\_seq)

**Output :**

****

**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 :**

def LongestRepeatingSubsequence(S):

    n = len(S)

    dp = [[0]\*(n+1) for \_ in range(n+1)]

    for i in range(1, n+1):

        for j in range(1, n+1):

            if S[i-1] == S[j-1] and i != j:

                dp[i][j] = dp[i-1][j-1] + 1

            else:

                dp[i][j] = max(dp[i-1][j], dp[i][j-1])

    i, j = n, n

    lrs = []

    while i > 0 and j > 0:

        if S[i-1] == S[j-1] and i != j:

            lrs.append(S[i-1])

            i -= 1

            j -= 1

        elif dp[i-1][j] >= dp[i][j-1]:

            i -= 1

        else:

            j -= 1

    return dp[n][n], ''.join(reversed(lrs))

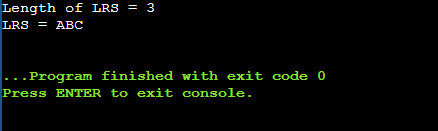
S = "AABCBDC"

length, lrs = LongestRepeatingSubsequence(S)

print("Length of LRS =", length)

print("LRS =", lrs)

**Output :**

****

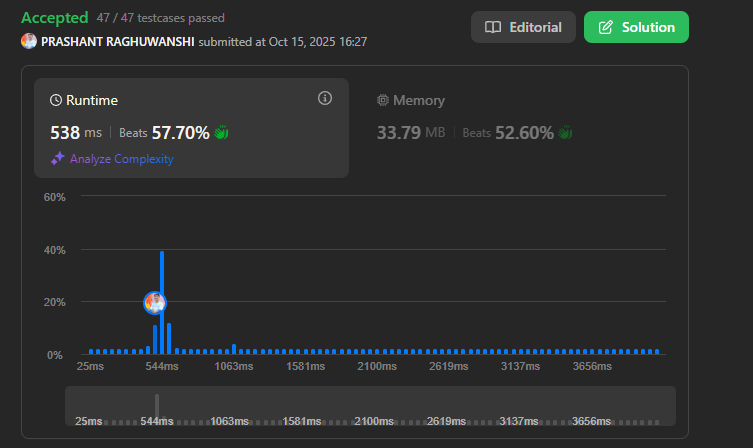
**LeetCode Assesment :**

**Problem Statement :**  <https://leetcode.com/problems/longest-common-subsequence/description/>

**Code :**

****

**Submission :**

****