Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 51
AAD Practical 8

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 8

Problem : A subsequence is a sequence that can be derived from another sequence by deleting some elements without changing the order of the remaining elements. Longest common subsequence (LCS) of 2 sequences is a subsequence, with maximal length, which is common to both the sequences.

Given two sequence of characters, P = M, N, O, M and Q = M, L, N, O, M, find any one longest common subsequence.

In case multiple solutions exist, print any of them. It is guaranteed that at least one non-empty common subsequence will exist.

Code :

```
import YSL_io


def LCS(P, Q):
s1, s2 = len(P), len(Q)
dp = [[0] * (s2 + 1) for _ in range(s1 + 1)]


for i in range(1, s1 + 1):
for j in range(1, s2 + 1):
if P[i - 1] == Q[j - 1]:
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 51
AAD Practical 8

```
dp[i][j] = dp[i - 1][j - 1] + 1
else:
dp[i][j] = max(dp[i - 1][j], dp[i][j - 1])

i, j = s1, s2
lcs = []

while i > 0 and j > 0:
if P[i - 1] == Q[j - 1]:
lcs.append(P[i - 1])
i -= 1
j -= 1
elif dp[i - 1][j] > dp[i][j - 1]:
i -= 1
else:
j -= 1

return ''.join(str(y) for y in lcs[::-1])

# Demo inputs :
# P = "MNOAM"
# Q = "MLNOM"

P=YSL_io.inputCYN("\n\tEnter the first sequence P : ")
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 51
AAD Practical 8

```
Q=YSL_io.inputCYN("\tEnter the second sequence Q : ")

YSL_io.printORNG('\n\tLongest Common Subsequence (LCS) : ', end='')

YSL_io.printRED(LCS(P, Q))
```

Screenshot :