

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1Solution 2Solution 3Solution 4

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 using namespace std;
5
6 vector<char> buildSequence(vector<vector<int>> lengths, string str);
7
8 // O(nm) time | O(nm) space
9 vector<char> longestCommonSubsequence(string str1, string str2) {
10     vector<vector<int>> lengths(str2.length() + 1,
11                               vector<int>(str1.length() + 1, 0));
12     for (int i = 1; i < str2.length() + 1; i++) {
13         for (int j = 1; j < str1.length() + 1; j++) {
14             if (str2[i - 1] == str1[j - 1]) {
15                 lengths[i][j] = lengths[i - 1][j - 1] + 1;
16             } else {
17                 lengths[i][j] = max(lengths[i - 1][j], lengths[i][j - 1]);
18             }
19         }
20     }
21     return buildSequence(lengths, str1);
22 }
23
24 vector<char> buildSequence(vector<vector<int>> lengths, string str) {
25     vector<char> sequence;
26     int i = lengths.size() - 1;
27     int j = lengths[0].size() - 1;
28     while (i != 0 && j != 0) {
29         if (lengths[i][j] == lengths[i - 1][j]) {
30             i--;
31         } else if (lengths[i][j] == lengths[i][j - 1]) {
32             j--;
33         } else {
34             sequence.insert(sequence.begin(), str[j - 1]);
35             i--;
36             j--;
37         }
38     }
39     return sequence;
40 }
41
```