```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(nm) time | O(nm) space
  public static List<Character> longestCommonSubsequence(String str1, String str2) {
    int[][] lengths = new int[str2.length() + 1][str1.length() + 1];
    for (int i = 1; i < str2.length() + 1; i++) {
      for (int j = 1; j < str1.length() + 1; j++) {
        if (str2.charAt(i - 1) == str1.charAt(j - 1)) {
          lengths[i][j] = lengths[i - 1][j - 1] + 1;
        } else {
          lengths[i][j] = Math.max(lengths[i - 1][j], lengths[i][j - 1]);
        }
      }
    }
    return buildSequence(lengths, str1);
  }

  public static List<Character> buildSequence(int[][] lengths, String str) {
    List<Character> sequence = new ArrayList<Character>();
    int i = lengths.length - 1;
    int j = lengths[0].length - 1;
    while (i != 0 && j != 0) {
      if (lengths[i][j] == lengths[i - 1][j]) {
        i--;
      } else if (lengths[i][j] == lengths[i][j - 1]) {
        j--;
      } else {
        sequence.add(0, str.charAt(j - 1));
        i--;
        j--;
      }
    }
    return sequence;
  }
}

```