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

#include <vector>
using namespace std;

// O(nm*min(n, m)) time | O((min(n, m))^2) space
vector<char> longestCommonSubsequence(string str1, string str2) {
  string small = str1.length() < str2.length() ? str1 : str2;
  string big = str1.length() >= str2.length() ? str1 : str2;
  vector<vector<char>> evenLcs;
  vector<vector<char>> oddLcs;
  for (int i = 0; i < small.length() + 1; i++) {
    evenLcs.push_back(vector<char>());
  }
  for (int i = 0; i < small.length() + 1; i++) {
    oddLcs.push_back(vector<char>());
  }
  for (int i = 1; i < big.length() + 1; i++) {
    vector<vector<char>> *currentLcs;
    vector<vector<char>> *previousLcs;
    if (i % 2 == 1) {
      currentLcs = &oddLcs;
      previousLcs = &evenLcs;
    } else {
      currentLcs = &evenLcs;
      previousLcs = &oddLcs;
    }
    for (int j = 1; j < small.length() + 1; j++) {
      if (big[i - 1] == small[j - 1]) {
        vector<char> copy = previousLcs->at(j - 1);
        copy.push_back(big[i - 1]);
        currentLcs->at(j) = copy;
      } else {
        currentLcs->at(j) =
            previousLcs->at(j).size() > currentLcs->at(j - 1).size()
                ? previousLcs->at(j)
                : currentLcs->at(j - 1);
      }
    }
  }
  return big.length() % 2 == 0 ? evenLcs[small.length()]
                               : oddLcs[small.length()];
}

```