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

using namespace std;

bool areInterwoven(string one, string two, string three, int i, int j,
                   vector<vector<int>> &cache);

// O(nm) time | O(nm) space - where n is the length of the
// first string and m is the length of the second string
bool interweavingStrings(string one, string two, string three) {
  if (three.size() != one.size() + two.size()) {
    return false;
  }

  vector<vector<int>> cache;
  for (int i = 0; i < one.size() + 1; i++) {
    cache.push_back(vector<int>{});
    for (int j = 0; j < two.size() + 1; j++) {
      cache[i].push_back(-1);
    }
  }

  return areInterwoven(one, two, three, 0, 0, cache);
}

bool areInterwoven(string one, string two, string three, int i, int j,
                   vector<vector<int>> &cache) {
  if (cache[i][j] != -1)
    return cache[i][j];

  int k = i + j;
  if (k == three.size())
    return true;

  if (i < one.size() && one[i] == three[k]) {
    cache[i][j] = areInterwoven(one, two, three, i + 1, j, cache);
    if (cache[i][j] == true)
      return true;
  }

  if (j < two.size() && two[j] == three[k]) {
    cache[i][j] = areInterwoven(one, two, three, i, j + 1, cache);
    return cache[i][j];
  }

  cache[i][j] = false;
  return false;
}
```