Our Solution(s) Run Code

Solution 1

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

// O(n * m^2 + nlog(n)) time | O(nm) space - where n is the number of
// m is the length of the longest string
function longestStringChain(strings) {
  // For every string, imagine the longest string chain that starts w
  // Set up every string to point to the next string in its respectiv
  // string chain. Also keep track of the lengths of these longest str
  const stringChains = {};
  for (const string of strings) {
    stringChains[string] = {nextString: '', maxChainLength: 1};
  }

  // Sort the strings based on their length so that whenever we visit
  // string (as we iterate through them from left to right), we can
  // already have computed the longest string chains of any smaller st
  const sortedStrings = strings.sort((a, b) => a.length - b.length);
  for (const string of sortedStrings) {
    findLongestStringChain(string, stringChains);
  }

  return buildLongestStringChain(strings, stringChains);
}

function findLongestStringChain(string, stringChains) {
  // Try removing every letter of the current string to see if the
  // remaining strings form a string chain.
  for (let i = 0; i < string.length; i++) {
    const smallerString = getSmallerString(string, i);
    if (!(smallerString in stringChains)) continue;
    tryUpdateLongestStringChain(string, smallerString, stringChains);
  }
}
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
function longestStringChain(strings) {
  // Write your code here.
}

// Do not edit the line below.
exports.longestStringChain = longestStringChain;

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.