Our Solution(s) Run Code

Solution 1 Solution 2

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

#include <vector>
#include <numeric>
#include <algorithm>

using namespace std;

// O(w * n * log(n) + n * w * log(w)) time | O(wn) space - where w is the number
// of words and n is the length of the longest word
vector<vector<string>> groupAnagrams(vector<string> words) {
  if (words.size() == 0)
    return {};

  vector<string> sortedWords = {};
  for (auto word : words) {
    sort(word.begin(), word.end());
    sortedWords.push_back(word);
  }

  vector<int> indices(words.size());
  iota(indices.begin(), indices.end(), 0);
  sort(indices.begin(), indices.end(), [sortedWords](int a, int b) -> bool {
    return sortedWords[a] < sortedWords[b];
  });

  vector<vector<string>> result = {};
  vector<string> currentAnagramGroup = {};
  string currentAnagram = sortedWords[indices[0]];
  for (auto index : indices) {
    string word = words[index];
    string sortedWord = sortedWords[index];

    if (sortedWord == currentAnagram) {
      currentAnagramGroup.push_back(word);
      continue;
    }

    result.push_back(currentAnagramGroup);
    currentAnagramGroup = vector<string>{word};
    currentAnagram = sortedWord;
  }

  result.push_back(currentAnagramGroup);

  return result;
}

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>

using namespace std;

vector<vector<string>> groupAnagrams(vector<string> words) {
  // Write your code here.
  return {};
}

```

Custom Output Raw Output Submit Code

Run or submit code when you're ready.