

Our Solution(s)Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 #include <numeric>
5 #include <algorithm>
6
7 using namespace std;
8
9 // O(w * n * log(n) + n * w * log(w)) time | O(w*n) space - where w is the number
10 // of words and n is the length of the longest word
11 vector<vector<string>> groupAnagrams(vector<string> words) {
12     if (words.size() == 0)
13         return {};
14
15     vector<string> sortedWords = {};
16     for (auto word : words) {
17         sort(word.begin(), word.end());
18         sortedWords.push_back(word);
19     }
20
21     vector<int> indices(words.size());
22     iota(indices.begin(), indices.end(), 0);
23     sort(indices.begin(), indices.end(), [sortedWords](int a, int b) {
24         return sortedWords[a] < sortedWords[b];
25     });
26
27     vector<vector<string>> result = {};
28     vector<string> currentAnagramGroup = {};
29     string currentAnagram = sortedWords[indices[0]];
30     for (auto index : indices) {
31         string word = words[index];
32         string sortedWord = sortedWords[index];
33     }
```

Your SolutionsRun Code

Solution 1Solution 2Solution 3

```
1 #include <vector>
2
3 using namespace std;
4
5 vector<vector<string>> groupAnagrams(vector<string> words) {
6     // Write your code here.
7     return {};
8 }
9
```

