Prompt | Scratchpad | Our Solution(s) | Video Explanation | Run Code

Solution 1 | Solution 2

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

using System;
using System.Linq;
using System.Collections.Generic;

public class Program {
  // O(w * n * log(n) + n * w * log(w)) time | O(wn) space - where w is the numbe
  // n is the length of the longest word
  public static List<List<string> > groupAnagrams(List<string> words) {
    if (words.Count == 0) return new List<List<string> >();

    List<string> sortedWords = new List<string>();
    foreach (string word in words) {
      char[] charArray = word.ToCharArray();
      Array.Sort(charArray);
      string sortedWord = new String(charArray);
      sortedWords.Add(sortedWord);
    }

    List<int> indices = Enumerable.Range(0, words.Count).ToList();
    indices.Sort((a, b) => sortedWords[a].CompareTo(sortedWords[b]));

    List<List<string> > result = new List<List<string> >();
    List<string> currentAnagramGroup = new List<string>();
    string currentAnagram = sortedWords[indices[0]];
    foreach (int index in indices) {
      string word = words[index];
      string sortedWord = sortedWords[index];

      if (sortedWord.Equals(currentAnagram)) {
        currentAnagramGroup.Add(word);
        continue;
      }

      result.Add(currentAnagramGroup);
      currentAnagramGroup = new List<string>(){
        word
      };
      currentAnagram = sortedWord;
    }

    result.Add(currentAnagramGroup);

    return result;
  }
}

```

Your Solutions | Run Code

Solution 1 | Solution 2 | Solution 3

```
using System.Collections.Generic;

class Program {
  public static List<List<string> > groupAnagrams(List<string> words) {
    // Write your code here.
    return null;
  }
}

```

Custom Output | Raw Output | Submit Code

Run or submit code when you're ready.