Problem | Editorial | Submissions

Print Anagrams Together

Difficulty: **Medium** Accuracy: **65.78%** Submissions: **93K+** Points: **4**

Given an array of strings, return all groups of strings that are anagrams. The strings in each group must be arranged in the order of their appearance in the original array. Refer to the sample case for clarification.

Examples:

Input: arr[] = ["act", "god", "cat", "dog", "tac"]
Output: [["act", "cat", "tac"], ["god", "dog"]]
Explanation: There are 2 groups of anagrams "god", "dog" make group 1. "act", "cat", "tac" make group 2.

Input: arr[] = ["no", "on", "is"]
Output: [["is"], ["no", "on"]]
Explanation: There are 2 groups of anagrams "is" makes group 1. "no", "on" make group 2.

Input: arr[] = ["listen", "silent", "enlist", "abc", "cab", "bac", "rat", "tar", "art"]

Java (1.8) | Average Time: 30m | Start Timer

```
1  // } Driver Code Ends
6  class Solution {
7      public ArrayList<ArrayList<String>> anagrams(String[] arr) {
8          Map<String, List<String>> anagramMap = new HashMap<>();
9          for (String str : arr) {
10             char[] charArray = str.toCharArray();
11             Arrays.sort(charArray);
12             String sortedStr = new String(charArray);
13             anagramMap.putIfAbsent(sortedStr, new ArrayList<>());
14             anagramMap.get(sortedStr).add(str);
15         }
16         ArrayList<ArrayList<String>> result = new ArrayList<>();
17         for (List<String> group : anagramMap.values()) {
18             result.add(new ArrayList<>(group));
19         }
20         return result;
21     }
22 }
23
24 // } Driver Code Ends
```

Custom Input | Compile & Run | Submit