49. Group Anagrams

09 February 2022 06:29 PM

Given an array of strings `strs`, group **the anagrams** together. You can return the answer in **any order**.

An **Anagram** is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Example 1:

```
Input: strs = ["eat","tea","tan","ate","nat","bat"]
Output: [["bat"],["nat","tan"],["ate","eat","tea"]]
```

Example 2:

```
Input: strs = [""]
Output: [[""]]
```

Example 3:

```
Input: strs = ["a"]
Output: [["a"]]
```

Constraints:

- `1 <= strs.length <= 10^4`
- `0 <= strs[i].length <= 100`
- `strs[i]` consists of lowercase English letters.

Let arr be

abcc	a1	b1	c2	
acbc	a1	b1	c2	
badc	a1	b1	c1	d1
abcd	a1	b1	c1	d1
dabb	a1	b2		d1
babd	a1	b2		d1
bdca	a1	b1	c1	d1
cabc	a1	b1	c2	

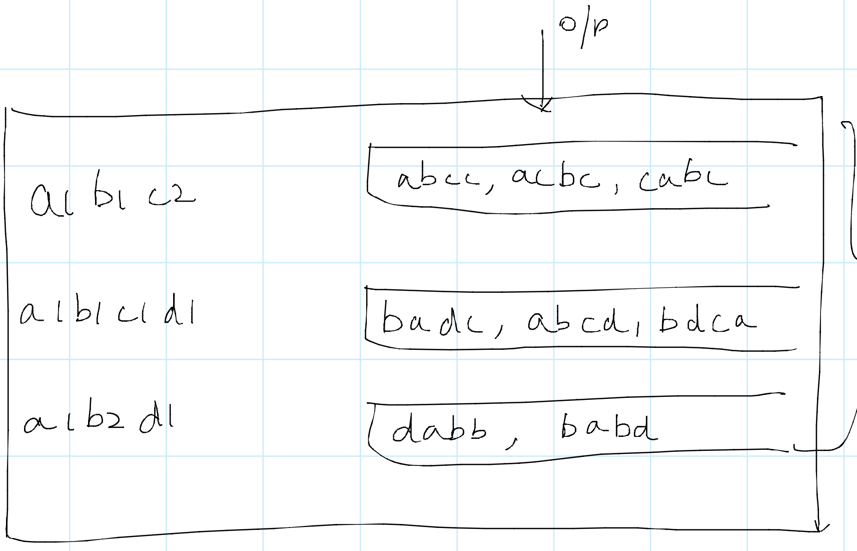
Anagrams are those whose frequencies value of element are same

o/p

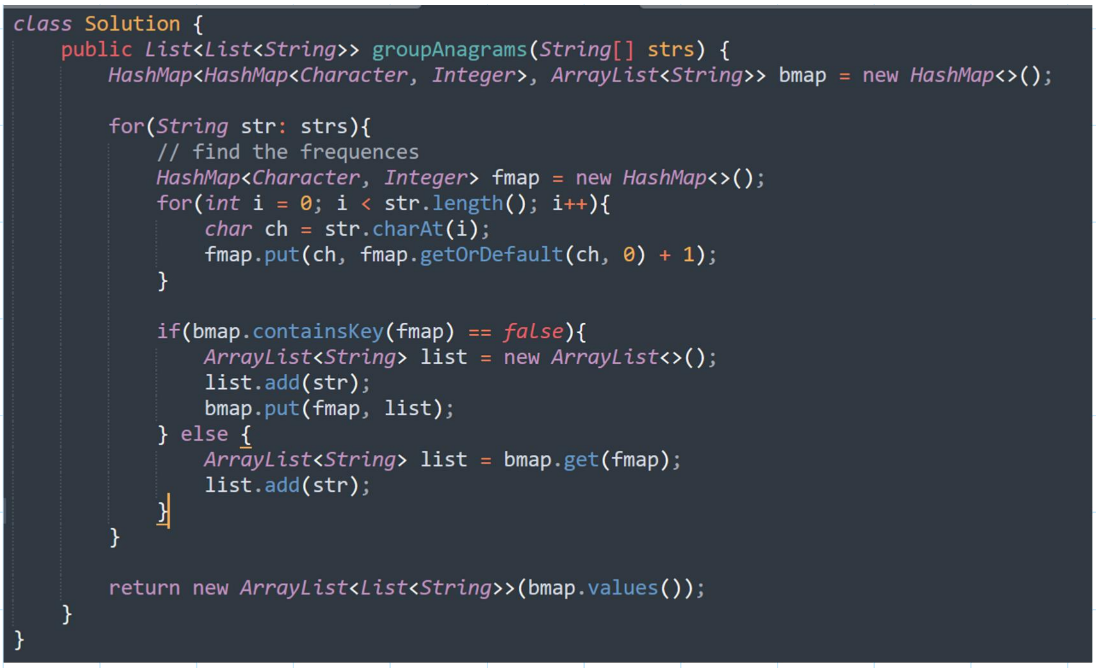
([abcc, acbc, cabc], [badc, abcd, dbca], [dabb, babd]]

we will use hashmap of frequences map.

$$HM < \underbrace{HM < c, i >}_{\text{Key}} , \underbrace{AL < s >}_{\text{value}} >$$



take this value and Save in Array list.



```java
class Solution {
    public List<List<String>> groupAnagrams(String[] strs) {
        HashMap<HashMap<Character, Integer>, ArrayList<String>> bmap = new HashMap<>();

        for(String str: strs){
            // find the frequences
            HashMap<Character, Integer> fmap = new HashMap<>();
            for(int i = 0; i < str.length(); i++){
                char ch = str.charAt(i);
                fmap.put(ch, fmap.getOrDefault(ch, 0) + 1);
            }

            if(bmap.containsKey(fmap) == false){
                ArrayList<String> list = new ArrayList<>();
                list.add(str);
                bmap.put(fmap, list);
            } else {
                ArrayList<String> list = bmap.get(fmap);
                list.add(str);
            }
        }

        return new ArrayList<List<String>>(bmap.values());
    }
}
```