

Problem: Letter Combinations of a Phone Number

Student: Ovidiu Mura

Description

Solution

Submissions

Discuss (999+)

Success

Details >

Runtime: 1 ms, faster than 78.56% of Java online submissions for Letter Combinations of a Phone Number.

Memory Usage: 38.9 MB, less than 6.16% of Java online submissions for Letter Combinations of a Phone Number.

Next challenges:

[Generate Parentheses](#)[Combination Sum](#)[Binary Watch](#)

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
a few seconds ago	Accepted	1 ms	38.9 MB	java
4 hours ago	Wrong Answer	N/A	N/A	java
4 hours ago	Wrong Answer	N/A	N/A	java

Java

Autocomplete

```
1 * class Solution {
2 *     public List<String> letterCombinations(String digits) {
3 *         Map<Integer, String> pad = new HashMap<>();
4 *         pad.put(2, "abc");
5 *         pad.put(3, "def");
6 *         pad.put(4, "ghi");
7 *         pad.put(5, "jkl");
8 *         pad.put(6, "mno");
9 *         pad.put(7, "pqrs");
10 *        pad.put(8, "tuv");
11 *        pad.put(9, "wxyz");
12 *        List<String> combinations = new ArrayList<>();
13 *        if(digits.length() == 0)
14 *            return combinations;
15 *        char[] cs = digits.toCharArray();
16 *        List<String> lettersGrp = new ArrayList();
17 *        for(char c: cs){
18 *            lettersGrp.add(pad.get(Character.getNumericValue(c)));
19 *        }
20 *        combinations.add("");
21 *        for (int ii=0; ii<lettersGrp.size(); ii++) {
22 *            ArrayList<String> str = new ArrayList<>();
23 *            for (int i = 0; i < combinations.size(); i++) {
24 *                for (int j = 0; j < lettersGrp.get(ii).length(); j++) {
25 *                    str.add(combinations.get(i) +
26 *                        Character.toString(lettersGrp.get(ii).toCharArray()[j]));
27 *                }
28 *            }
29 *            combinations.clear();
30 *            combinations.addAll(str);
31 *        }
32 *        return combinations;
33 *    }
34 * }
```

```
class Solution {
    public List<String> letterCombinations(String digits) {
        Map<Integer, String> pad = new HashMap<>();
        pad.put(2, "abc");
        pad.put(3, "def");
        pad.put(4, "ghi");
        pad.put(5, "jkl");
        pad.put(6, "mno");
        pad.put(7, "pqrs");
        pad.put(8, "tuv");
        pad.put(9, "wxyz");
        List<String> combinations = new ArrayList<>();
        if(digits.length() == 0)
            return combinations;
        char[] cs = digits.toCharArray();
        List<String> lettersGrp = new ArrayList();
        for(char c: cs){
            lettersGrp.add(pad.get(Character.getNumericValue(c)));
        }
        combinations.add("");
        for (int ii=0; ii<lettersGrp.size(); ii++) {
            ArrayList<String> str = new ArrayList<>();
            for (int i = 0; i < combinations.size(); i++) {
                for (int j = 0; j < lettersGrp.get(ii).length(); j++) {
                    str.add(combinations.get(i) +
                        Character.toString(lettersGrp.get(ii).toCharArray()[j]));
                }
            }
        }
    }
}
```

```
    }  
    combinations.clear();  
    combinations.addAll(strs);  
  }  
  return combinations;  
}  
}
```