784 Letter Case Permutation

2018年5月4日 19:12

Given a string S, we can transform every letter individually to be lowercase or uppercase to create another string. Return a list of all possible strings we could create.

Examples:

Input: S = "a1b2"

Output: ["a1b2", "a1B2", "A1b2", "A1B2"]

Input: S = "3z4"

Output: ["3z4", "3Z4"]

Input: S = "12345"

Output: ["12345"]

Note:

- S will be a string with length at most 12.
- S will consist only of letters or digits.

来自 <https://leetcode.com/problems/letter-case-permutation/description/>

给定一个字符串S，通过将字符串S中的每个字母转变大小写，我们可以获得一个新的字符串。返回所有可能得到的字符串集合。

示例:

输入: S = "a1b2"

输出: ["a1b2", "a1B2", "A1b2", "A1B2"]

输入: S = "3z4"

输出: ["3z4", "3Z4"]

输入: S = "12345"

输出: ["12345"]

注意:

- S 的长度不超过12。
- S 仅由数字和字母组成。

Solution for Python3:

```python
class Solution1:
    def letterCasePermutation(self, S):
        """
        :type S: str
        :rtype: List[str]
        """
        if not S:
            return [""]
        ans = []
        self.DFS(S, ans, 0)
        return ans

    def DFS(self, s, ans, pos):
        if pos == len(s):
            ans.append(s)
            return
        if s[pos].isdigit():
            self.DFS(s, ans, pos+1)
            return
        s1 = copy.deepcopy(s)
        s1 = s1[:pos] + s1[pos].swapcase() + s1[pos+1:]
```

```
        self.DFS(s, ans, pos + 1)
        self.DFS(s1, ans, pos + 1)


# S = "a1b2"
# L = [['a', 'A'], '1', ['b', 'B'], '2']
# 解包*L = ['a','A'],'1',['b','B'],'2'
# itertools.produce():笛卡尔积，括号中每一部分均拿出一项与其他部分组合
# ('a', '1', 'b', '2')
# ('a', '1', 'B', '2')
# ('A', '1', 'b', '2')
# ('A', '1', 'B', '2')
class Solution2:
    def letterCasePermutation(self, S):
        """
        :type S: str
        :rtype: List[str]
        """
        L = [[i.lower(), i.upper()] if i.isalpha() else i for i in S]
        return [''.join(i) for i in itertools.product(*L)]
```

Solution for C++:

```
class Solution {
public:
    vector<string> letterCasePermutation(string S) {
        vector<string> res;
        DFS(S, 0, res);
        return res;
    }

    void DFS(string& s, int pos, vector<string>& res) {
        if (pos == s.length()) {
            res.push_back(s);
            return;
        }
        if (isdigit(s[pos])) {
            DFS(s, pos+1, res);
            return;
        }
        DFS(s, pos+1, res);
        // toggle case
        s[pos] ^= (1 << 5);
        DFS(s, pos+1, res);
    }
};
```

Appendix:

C++ 大小写转换: s ^= (1<<5)

Python 大小写转换: chr(ord(s) ^ (1<<5)) or s.swapcase()

Python itertools.product(*L):笛卡尔积

L=[[1,2],3,[4,5]]

***L 会解包大概生成[1,2],3,[4,5]**

笛卡尔积会组合每一项:

1,3,4
1,3,5
2,3,4
2,3,5