0017. 电话号码的字母组合

ITCharge 大约 2 分钟

- 标签：哈希表、字符串、回溯
- 难度：中等

题目链接

- 0017. 电话号码的字母组合 - 力扣

题目大意

描述：给定一个只包含数字 2~9 的字符串 `digits` 。给出数字到字母的映射如下（与电话按键相同）。注意 1 不对应任何字母。

要求：返回字符串 `digits` 在九宫格键盘上所能表示的所有字母组合。答案可以按 「任意顺序」返回。

说明：

- $0 \leq digits.length \leq 4$。
- `digits[i]` 是范围 $2 \sim 9$ 的一个数字。

示例：

- 示例 1：

```py
输入：digits = "23"
输出：["ad","ae","af","bd","be","bf","cd","ce","cf"]
```

- 示例 2：

```py
输入：digits = "2"
输出：["a","b","c"]
```

解题思路

思路 1：回溯算法 + 哈希表

用哈希表保存每个数字键位对应的所有可能的字母，然后进行回溯操作。

回溯过程中，维护一个字符串 combination，表示当前的字母排列组合。初始字符串为空，每次取电话号码的一位数字，从哈希表中取出该数字所对应的所有字母，并将其中一个插入到 combination 后面，然后继续处理下一个数字，知道处理完所有数字，得到一个完整的字母排列。开始进行回退操作，遍历其余的字母排列。

思路 1：代码

```py
class Solution:
    def letterCombinations(self, digits: str) -> List[str]:
        if not digits:
            return []

        phone_dict = {
            "2": "abc",
            "3": "def",
```

```
        "4": "ghi",
        "5": "jkl",
        "6": "mno",
        "7": "pqrs",
        "8": "tuv",
        "9": "wxyz"
    }

    def backtrack(combination, index):
        if index == len(digits):
            combinations.append(combination)
        else:
            digit = digits[index]
            for letter in phone_dict[digit]:
                backtrack(combination + letter, index + 1)

    combinations = list()
    backtrack('', 0)
    return combinations
```

思路 1：复杂度分析

- **时间复杂度**：$O(3^m \times 4^n)$，其中 r `digits` 中对应 3 个字母的数字个数，m 是 `digits` 中对应 4 个字母的数字个数。
- **空间复杂度**：$O(m + n)$。