LeetCode17:Letter Combinations of a Phone Number

题目：

Given a digit string, return all possible letter combinations that the number could represent.

A mapping of digit to letters (just like on the telephone buttons) is given below.


```
Input:Digit string "23"
Output: ["ad", "ae", "af", "bd", "be", "bf", "cd", "ce", "cf"].
```

Note:

Although the above answer is in lexicographical order, your answer could be in any order you want.

这道题牛b的人使用递归来解显得b格糕。但我比较笨，就用笨办法吧。就是最好理解的挨个历遍，挨个字符concatenate
代码是python

```
3  def letterCombinations(self, digits):
4    dic = {
5      '2': ['a', 'b', 'c'],
6      '3': ['d', 'e', 'f'],
7      '4': ['g','h','i'],
8      '5': ['j','k','l'],
9      '6': ['m','n','o'],
10     '7': ['p','q','r','s'],
11     '8': ['t','u','v'],
12     '9': ['w','x','y','z']
13     }
14   if not digits:
15     return []
16
17   result = ['']
18   for char in digits:
19     letterList = dic[char]
20     temp = []
21     for letter in letterList:
22       for res in result:
23         temp.append(res + letter)
24     result = temp
25
26   return result
27
28
29 #here is the testing code
30 result = LetterCombination("23")
31 print(result)
```

第一步： 用一个字典保存2-9所有的数字代表的字符。
第二步： 判断输入的是否是数字组成的字符串，如果不是或者是空字符穿则返回一个空的list (line14-15)

第三步： 建立一个初始化的list用于存放结果 result = [''], 注意到这里我把它初始化为只有一个空字符的list, 而不是一个空的list. 下面会解释为什么这么做。

第四步： 开始进行历遍。line 18 - line 24
首先历遍输入的数字字符串，例如对于“234”就历遍‘2’, ‘3’, 和‘4’。
对于每一个字符串中的字符，从字典中获取它相对应的字母字符串数组。
例如 对于‘2’ ， 则从字典中取出字符 [‘a’, ‘b’, ‘c’]
然后进而历遍这个字符数组， 与结果result 中的每一个字符相加(concatenation)，然后用获得的结果更新result。其中还要对结果中的元素进行一次历遍，因此有三个for loop.
这就是为什么result 需要初始化为 [''] 而不是[], 必须由东西在里面才能进行历遍。

复杂度：
假设由数字由 n 个， 每个数字对应 k 个字母， 则时间复杂度为O(n^k)