CSX3009

Algorithm Design

2/2021

LECTURER

THITIPONG TANPRASERT

Letter Combination of a Phone Number

Difficulty: Medium

Leetcode.com/letter-combinations-of-a-phone-number

Table of Contents

I Problem Statement

- Given a string containing digits from 2-9 inclusive, return all possible letter combinations that the number could represent. Return the answer in any order.

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

**NOTE **

- The digit 0 maps to 0 itself.
- The digit 1 maps to 1 itself.

Input / Output

EXAMPLE 1:

Input : digits = "23"
Output : ["ad","ae","af","bd","be","bf","cd","ce","cf"]

EXAMPLE 2:

Input : digits = ""
Output : []

EXAMPLE 3:

Input : digits = "2"
Output : ["a","b","c"]

**Constraints **

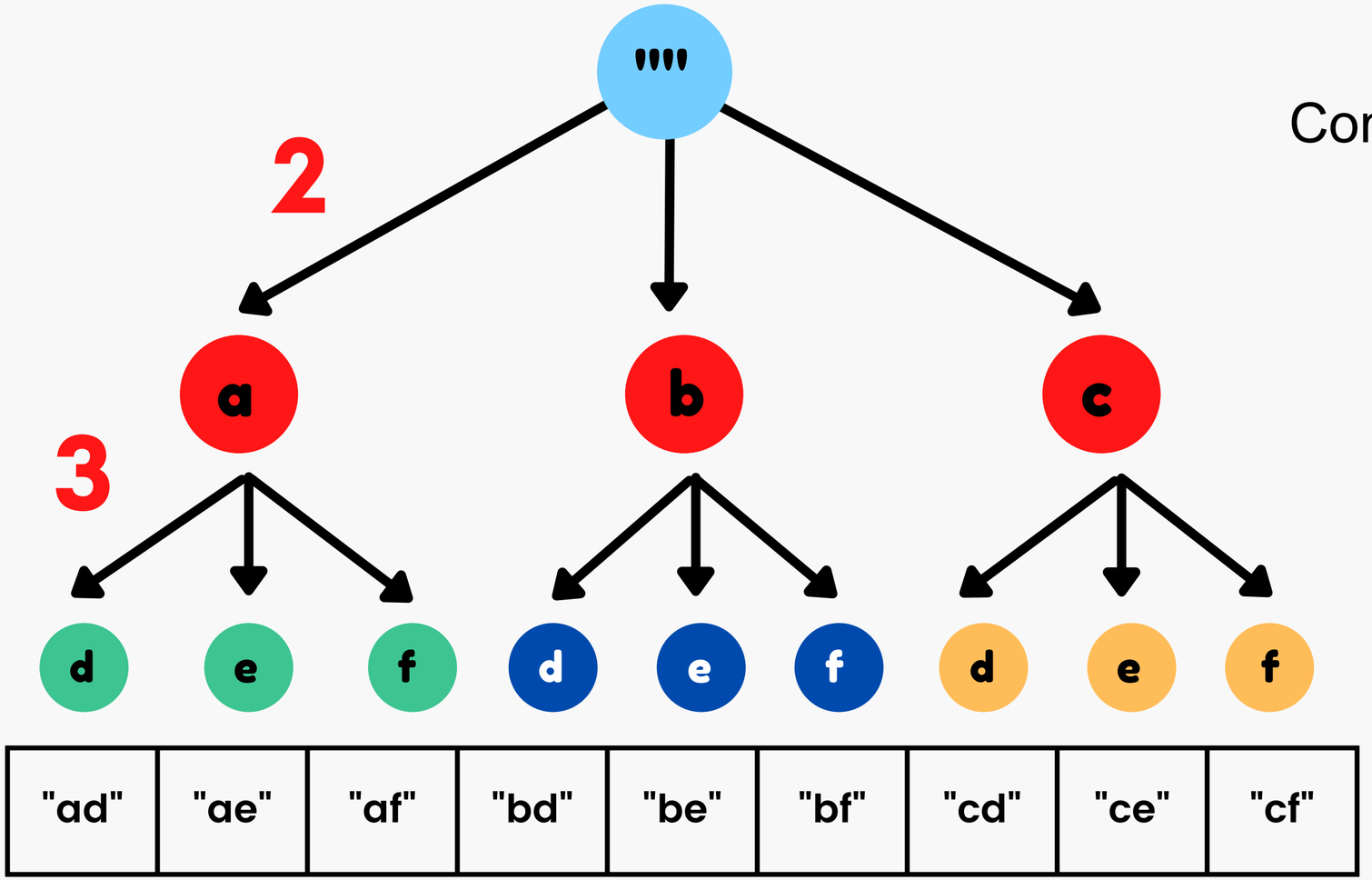
- 0 <= digits.length <= 4
- digits[i] is a digit in the range ['2', '9']

Algorithm That We Use

BackTracking Recursion

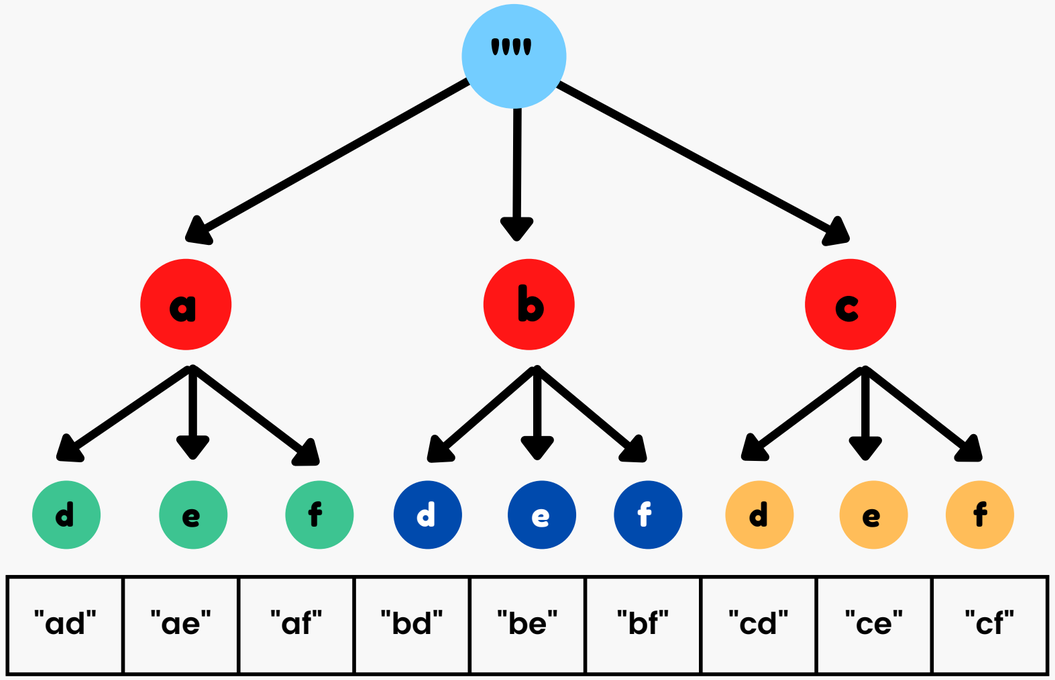
The problem can be solved using the backtracking approach. The idea is to consider a digit as the starting point and generate all possible combinations with that letter.

Input : digits = "23"
Output : ["ad","ae","af","bd","be","bf","cd","ce","cf"]



Combination possible:
3*3 = 9

II Algorithm



THERE IS ONE OBSERVATION THAT WHEN WE HAVE TWO DIGITS EACH WITH 3 POSSIBLE CHARACTER REPRESENTATION (E.G. "23") THEN NO. OF POSSIBLE COMBINATIONS = 3^2. WHEN THERE IS ONE DIGIT REPRESENTING 3 ALPHABETS AND ONE DIGIT REPRESENTING 4 ALPHABETS, THEN NO. OF POSSIBLE COMBINATIONS = 3*4

Pseudocode

- If the input is empty, simply return an empty array.

- Initialize a hashmap that maps digits to their letters, i.e. mapping “2” to “a”, “b”, and “c”.

```
phone_letter = {
"2" : "abc",
"3" : "def",
"4" : "ghi",
"5" : "jkl",
"6" : "mno",
"7" : "qprs",
"8" : "tuv",
"9" : "wxyz"
}
```

Pseudocode

1. CONSIDER THE PARAMETERS OF THE BACKTRACKING FUNCTION AS THE PATH, I.E THE CURRENT PATH WE ARE TRAVERSING ON AND THE INDEX, I.E. ON THE DIGIT WE ARE ON.

2. THE BASE CASE WOULD BE, IF OUR CURRENT COMBINATION OF LETTERS IS THE SAME LENGTH AS THE INPUT DIGITS, THAT ITERATION IS COMPLETED. THEREFORE, INSERT IT INTO THE ANSWER LIST, AND BACKTRACK.
ELSE, FIND ALL THE POSSIBLE COMBINATIONS OF LETTERS THAT CORRESPOND WITH THE CURRENT DIGIT I.E. DIGITS[INDEX].

3. TRAVERSE THROUGH THE LETTERS. FOR EACH LETTER, INSERT THE LETTER TO OUR CURRENT PATH, AND BACKTRACK AGAIN, AND INCREMENT THE INDEX BY 1.

4. REMOVE THE LETTER FROM THE PATH ONCE THE ITERATION IS COMPLETED.

Implementation

Python 3

```python
class Solution:
    def letterCombinations(self, digits: str) -> List[str]:
        result = []
        phone_dic = {
            "2" : "abc",
            "3" : "def",
            "4" : "ghi",
            "5" : "jkl",
            "6" : "mno",
            "7" : "qprs",
            "8" : "tuv",
            "9" : "wxyz"}

        def backtrack(i, currentString):
            if len(currentString) == len(digits):
                result.append(currentString)
                return
            for p in phone_dic[digits[i]]:
                backtrack(i + 1, currentString + p)

        if digits:
            backtrack(0, "")

        return result
```

Accepted Runtime: 32 ms

Your input	"23"
Output	["ad","ae","af","bd","be","bf","cd","ce","cf"]
Expected	["ad","ae","af","bd","be","bf","cd","ce","cf"]

Implementation

Python 3

```
def letterCombinations(self, digits: str) -> List[str]:
    result = []
    phone_dic = {
        "2" : "abc",
        "3" : "def",
        "4" : "ghi",
        "5" : "jkl",
        "6" : "mno",
        "7" : "qprs",
        "8" : "tuv",
        "9" : "wxyz"}
```

here we add "result" it will save the value.
after that we add a "phone_dic" first we add a digits that in range 2 - 9, and follow up dictionary in the example.

III Implementation

Python 3

```
def backtrack(i, currentString):
    if len(currentString) == len(digits):
        result.append(currentString)
        return
```

as you can see "i" going to tell what index at in digit string "currentString" is reading the dictionary.
after that len(currentString) is equally to len(digits)
what it means is it will take every single digits and map in currentStirng.

```
for p in phone_dic[digits[i]]:
    backtrack(i + 1, currentString + p)
```

```
"7" : "qprs",
"8" : "tuv",
"9" : "wxyz"}
```

in here, digits[i] will tell what digits we are at, and phone_dic[digit[i]] will help to map a 4 dictionary that i marked it in yellow

next, backtrack(i +1) to move to the next digits,
and (currentString + p) to give crrentString a visits to p

```
if digits:
    backtrack(0, "")

return result
```

Example 2:

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

lastly, if digits is empty the recursive call that is backtrack is 0 it will return as example 2 empty array.

Approach Submission

Running time: 48 ms, Memory: 14 MB

time: 48 ms, faster than 40.46% of Python3 online submissions for Letter Combinations of a Phone Number.

nory Usage: 14 MB, less than 57.43% of Python3 online submissions for Letter Combinations of a Phone Number.

t challenges:

enerate Parentheses | Combination Sum | Binary Watch

w off your acceptance:

me Submitted	Status	Runtime	Memory	Language
3/09/2022 21:57	Accepted	48 ms	14 MB	python3

References

6217429 VIBOLROTTANA SENG

6215106 Anwar Rasheed

YouTube:

https://www.youtube.com/watch?v=0snEunUacZY

LeetCode Dicussion:

https://leetcode.com/problems/letter-combinations-of-a-phone-number/discuss/?currentPage=1&orderBy=most_votes&query=

CSX3009

Algorithm Design

LECTURER

THITIPONG TANPRASERT

Thank you for listening!