75 Days of Code
Day 57
Problem no : Leetcode 875
Problem Title :Letter Combinations of a Phone Number
Problem type : recursion

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 digits to letters (just like on the telephone buttons) is given below. Note that 1 does not map to any letters.

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"]

```typescript
function letterCombinations(digits: string): string[] {
  if (!digits) return [];
  let result: string[] = [];

  const phoneNumberToLetters = {
    2: ["a", "b", "c"],
    3: ["d", "e", "f"],
    4: ["g", "h", "i"],
    5: ["j", "k", "l"],
    6: ["m", "n", "o"],
    7: ["p", "q", "r", "s"],
    8: ["t", "u", "v"],
    9: ["w", "x", "y", "z"],
  };

  const recursiveCall = (index: number, path: string) => {
    if (path.length === digits.length) {
      result.push(path);
      return;
    }

    const letters = phoneNumberToLetters[digits[index]];

    for (const letter of letters) {
      recursiveCall(index + 1, path + letter);
    }
  };
  recursiveCall(0, "");
  return result;
}
```

Accepted

Runtime	Details	Memory
57 ms		42.92 MB
Beats 43.75% of users with TypeScript		Beats 64.48% of users with TypeScript

More challenges