17. Letter Combinations of a Phone Number

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"]

**Constraints:**

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

**Brute Force**

Since the problem statement is asking for us to return **all possible** letter combinations, this makes me think it is a backtracking problem.

There are a couple ways we can do this.

We could do an iterative BFS traversal using a queue

We could do a recursive DFS traversal

If we were to draw out an execution tree, we would see that in the worst case, adding one more letter would increase the size and space of the problem by a factor of 4 (for number 7). Therefore, the time complexity is O(4^n).

The final output of the space complexity will take O(4^n) space.

**Iterative BFS**

To efficiently access the contents of each number, we will store the