**17. Letter Combinations of a Phone Number :-**

Medium Accepted: 1.6M Submissions: 2.8M Acceptance Rate: 57.6%

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'].

**Code :-**

class Solution {

public:

    void func(string &digits, vector<vector<char>> &mp, int d, vector<string> &ans, string &s){

        if(d == digits.size()){

            ans.push\_back(s);

            return;

        }

        for(auto &ch:mp[digits[d] - '0']){

            s.push\_back(ch);

            func(digits, mp, d+1, ans, s);

            s.pop\_back();

        }

        return;

    }

    vector<string> letterCombinations(string digits) {

        vector<string> ans;

        string s="";

        int d=digits.size();

        if(d == 0)        return ans;

        vector<vector<char>> mp(10);

        mp[2] = {'a','b','c'};      mp[3] = {'d','e','f'};

        mp[4] = {'g','h','i'};      mp[5] = {'j','k','l'};

        mp[6] = {'m','n','o'};      mp[7] = {'p','q','r','s'};

        mp[8] = {'t','u','v'};      mp[9] = {'w','x','y','z'};

        func(digits, mp, 0, ans, s);

        return ans;

    }

};