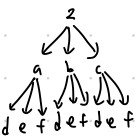
☆ Backtracking : It's a problem-solving algorithmic technique that involves finding a solution incrementally by trying different options and undoing them if they lead to a dead end. It's commonly used in situations where you need to explore multiple possibilities to solve a problem, like searching for a path.

Letter Combinations of a Phone Number

Ex) digits = "23"

Output: ["ad", "ae", "af", "bd","be","bf", "cd", "ce", "cf"]



Algorithm:

```
num_to_char = {"2": "abc",
               :
               "9": "wxyz"}

def backtrack(index, curstr):
    if index == len(digits):
        res.append(curstr)
        return

    for c in num_to_char[digits[index]]:
        backtrack(index+1, curstr + c)
```

① bt(0, "") → bt(1, a) , stack(bt(1,b), bt(1,c))

② bt(1,a) → bt(2, ad), stack(bt(2,ae), bt(2,af))

③ bt(2,ad)

④ bt(2,ae)

⑤ bt(2,af)

⑥ bt(1,b) → bt(2,bd) , stack(bt(2,be), bt(2,bf))

⋮