079. Word Search

079 Word Search

- **Backtracking**

Description

Given a 2D board and a word, find if the word exists in the grid.

The word can be constructed from letters of sequentially adjacent cell, where "adjacent" cells are those horizontally or vertically neighboring. The same letter cell may not be used more than once.

For example,
Given **board** =

```
[
  ['A','B','C','E'],
  ['S','F','C','S'],
  ['A','D','E','E']
]
```

word = `"ABCCED"`, -> returns `true`,
word = `"SEE"`, -> returns `true`,
word = `"ABCB"`, -> returns `false`.

1. Thought line

2. Backtracking

```
class Solution {
public:
    bool exist(vector<vector<char>>& board, string word) {
        if(board.size()==0 || board[0].size()==0 )
            return true;

        for(int i=0; i<board.size(); i++){
            for(int j=0; j<board[0].size(); j++){
                if(check(board, word, i, j))
                    return true;
            }
        }
        return false;
    }

    bool check(vector<vector<char>>& board, string word, int i, int j){
        if(word.length()==0)
            return true;
        if(i<0 || j<0 ||i>=board.size() ||j>=board[0].size())
            return false;
        if(word[0]==board[i][j]){
            char c = word[0];
            board[i][j]='\\0';
            if(check(board,word.substr(1), i+1, j)||
                check(board,word.substr(1), i-1, j)||
                check(board,word.substr(1), i, j+1)||
                check(board,word.substr(1), i, j-1))
                return true;
            board[i][j]=c;
        }
```

```
        return false;
    }
};
```