**Algorithm**

There is a certain code pattern for all the algorithms of backtracking. For example, one can find one template in our [Explore card of Recursion II](https://leetcode.com/explore/learn/card/recursion-ii/472/backtracking/2793/).

The skeleton of the algorithm is a loop that iterates through each cell in the grid. For each cell, we invoke the backtracking function (i.e. backtrack()) to check if we would obtain a solution, starting from this very cell.

For the backtracking function backtrack(row, col, suffix), as a DFS algorithm, it is often implemented as a recursive function. The function can be broke down into the following four steps:

* Step 1). At the beginning, first we check if we reach the bottom case of the recursion, where the word to be matched is empty, i.e. we have already found the match for each prefix of the word.
* Step 2). We then check if the current state is invalid, either the position of the cell is out of the boundary of the board or the letter in the current cell does not match with the first letter of the word.
* Step 3). If the current step is valid, we then start the exploration of backtracking with the strategy of DFS. First, we mark the current cell as visited, e.g. any non-alphabetic letter will do. Then we iterate through the four possible directions, namely up, right, down and left. The order of the directions can be altered, to one's preference.
* Step 4). At the end of the exploration, we revert the cell back to its original state. Finally we return the result of the exploration.