**目 录**

[1 基本数据结构 1](#_Toc93181012)

[2 二叉树 1](#_Toc93181013)

[3 回溯 1](#_Toc93181014)

[3.1 矩阵中的路径(单词搜索) 1](#_Toc93181015)

[3.2 机器人的运动范围 2](#_Toc93181016)

[4 动态规划 3](#_Toc93181017)

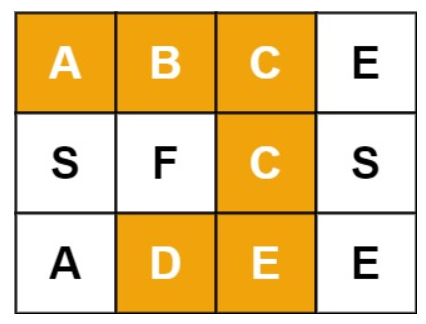
[4.1 剪绳子 3](#_Toc93181018)

1. 基本数据结构
2. 二叉树
3. 回溯
   1. 矩阵中的路径(单词搜索)

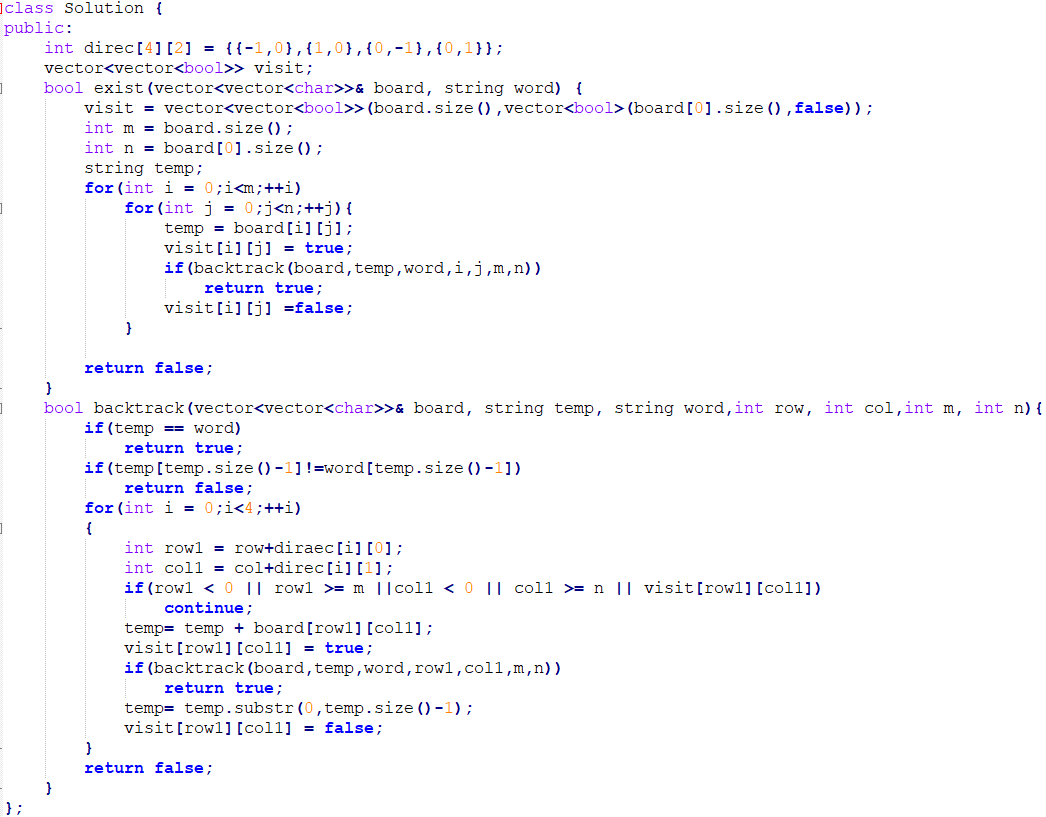
给定一个 m x n 二维字符网格 board 和一个字符串单词 word 。如果 word 存在于网格中，返回 true ；否则，返回 false 。

单词必须按照字母顺序，通过相邻的单元格内的字母构成，其中“相邻”单元格是那些水平相邻或垂直相邻的单元格。同一个单元格内的字母不允许被重复使用。

例如，在下面的 3×4 的矩阵中包含单词 "ABCCED"（单词中的字母已标出）



**【解题思路】**非常明显的回溯题，注意：1. 在主函数里面需要遍历二维数组获取单词起点，再进行回溯；2. 根据每一位字符进行剪枝，而不是每次判断整个单词是否相等。



* 1. 机器人的运动范围

地上有一个m行n列的方格，从坐标 [0,0] 到坐标 [m-1,n-1] 。一个机器人从坐标 [0, 0] 的格子开始移动，它每次可以向左、右、上、下移动一格（不能移动到方格外），也不能进入行坐标和列坐标的数位之和大于k的格子。例如，当k为18时，机器人能够进入方格 [35, 37] ，因为3+5+3+7=18。但它不能进入方格 [35, 38]，因为3+5+3+8=19。请问该机器人能够到达多少个格子？

**【解题思路】**回溯时需注意两个问题，1，递归不回溯，返回后visit不需要设回false；2，只需要向右向下移动即可，不需要遍历四个方向。



1. 动态规划
   1. 剪绳子

给你一根长度为 n 的绳子，请把绳子剪成整数长度的 m 段（m、n都是整数，n>1并且m>1），每段绳子的长度记为 k[0],k[1]...k[m-1] 。请问 k[0]\*k[1]\*...\*k[m-1] 可能的最大乘积是多少？例如，当绳子的长度是8时，我们把它剪成长度分别为2、3、3的三段，此时得到的最大乘积是18。

**【解题思路】**能想到是动态规划，取最大值时实际应该是：（共四项）因为m>1，其中(i-j)\*j表示只剪一刀，dp[i-j]\*dp[j]表示至少3刀，dp[i-j]\*j,和dp[j]\*[i-j]表示至少两刀。

max(dp[i-j]\*dp[j], (i-j)\*j, dp[i-j]\*j, dp[j]\*[i-j])

因为会进行迭代，所以有些像将是重复的，可以只保留其中两项。max(j\*dp[i-j],j\*(i-j))

