从递归入手三维动态规划

前置知识:
讲解068-从递归入手二维动态规划
从讲解066开始都是动态规划大专题，建议从头开始学习会比较容易理解

从递归到三维动态规划，包含多维费用背包
严格位置依赖的三维动态规划
三维动态规划的空间压缩

注意：
多维费用背包问题就是很普通的动态规划
但是【必备】课程里还会安排背包dp的内容，那时候会把其他几种背包问题做汇总讲述

左程云

从递归入手三维动态规划

尝试函数有1个可变参数可以完全决定返回值，进而可以改出1维动态规划表的实现
同理
尝试函数有2个可变参数可以完全决定返回值，那么就可以改出2维动态规划的实现
同理
尝试函数有3个可变参数可以完全决定返回值，那么就可以改出3维动态规划的实现

大体过程都是：
写出尝试递归
记忆化搜索(从顶到底的动态规划)
严格位置依赖的动态规划(从底到顶的动态规划)
空间、时间的更多优化

原理完全一样，可以参考讲解067　那么直接看题目吧！

左程云

从递归入手三维动态规划

题目1
一和零(多维费用背包)
给你一个二进制字符串数组 strs 和两个整数 m 和 n
请你找出并返回 strs 的最大子集的长度
该子集中 最多 有 m 个 0 和 n 个 1
如果 x 的所有元素也是 y 的元素，集合 x 是集合 y 的 子集
测试链接 : https://leetcode.cn/problems/ones-and-zeroes/

从递归入手三维动态规划

题目2
盈利计划(多维费用背包)
集团里有 n 名员工，他们可以完成各种各样的工作创造利润
第 i 种工作会产生 profit[i] 的利润，它要求 group[i] 名成员共同参与
如果成员参与了其中一项工作，就不能参与另一项工作
工作的任何至少产生 minProfit 利润的子集称为 盈利计划
并且工作的成员总数最多为 n
有多少种计划可以选择？因为答案很大，所以 返回结果模 10^9 + 7 的值。
测试链接：https://leetcode.cn/problems/profitable-schemes/

从递归入手三维动态规划

题目3
骑士在棋盘上的概率
n * n的国际象棋棋盘上，一个骑士从单元格(row, col)开始，并尝试进行 k 次移动
行和列从0开始，所以左上单元格是 (0,0)，右下单元格是 (n-1, n-1)
象棋骑士有8种可能的走法。每次移动在基本方向上是两个单元格，然后在正交方向上是一个单元格
每次骑士要移动时，它都会随机从8种可能的移动中选择一种，然后移动到那里
骑士继续移动，直到它走了 k 步或离开了棋盘
返回 骑士在棋盘停止移动后仍留在棋盘上的概率
测试链接：https://leetcode.cn/problems/knight-probability-in-chessboard/

从递归入手三维动态规划

题目4
矩阵中和能被 K 整除的路径
给一个下标从0开始的 n * m 整数矩阵 grid 和一个整数 k
从起点(0,0)出发，每步只能往下或者往右，你想要到达终点(m-1, n-1)
请你返回路径和能被 k 整除的路径数目
由于答案可能很大，返回答案对10^9+7取余的结果
测试链接：
https://leetcode.cn/problems/paths-in-matrix-whose-sum-is-divisible-by-k/

左程云

从递归入手三维动态规划

题目5
扰乱字符串
使用下面描述的算法可以扰乱字符串 s 得到字符串 t ：
步骤1：如果字符串的长度为 1 ，算法停止
步骤2：如果字符串的长度 > 1 ，执行下述步骤：
在一个随机下标处将字符串分割成两个非空的子字符串
已知字符串s，则可以将其分成两个子字符串x和y且满足s=x+y
可以决定是要 交换两个子字符串 还是要 保持这两个子字符串的顺序不变
即s可能是 s = x + y 或者 s = y + x
在x和y这两个子字符串上继续从步骤1开始递归执行此算法
给你两个 长度相等 的字符串 s1 和 s2，判断 s2 是否是 s1 的扰乱字符串
如果是，返回true ；否则，返回false
测试链接：https://leetcode.cn/problems/scramble-string/