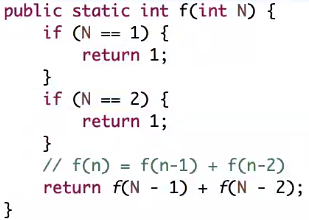
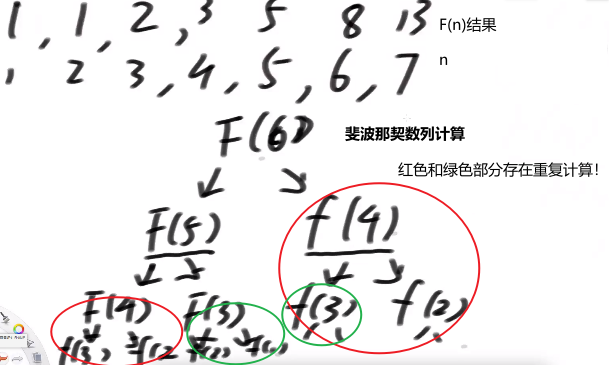
暴力递归：不记录每一个子问题的解；

动态规划： 记录每一个子问题的解。

动态规划是一种记忆化搜索，把参数组合映射到结构化的缓存；DP是结果，不是原因。

动态规划的状态转移跟暴力递归的尝试策略是一码事，DP都可以由暴力递归改写而来，但不是所有暴力递归都能够改成DP（没有重复的子问题）。

暴力递归重复计算优化：

找出递归函数的变化参数，

1. 对于字符串，如果无强依赖关系，可以增加hashmap等全局缓存表，使用记忆搜索；
2. 对于数字，如果不存在依赖关系，可以增加数组等全局缓存表，使用记忆搜索；

3. 对于数字，如果存在严格位置依赖，可以使用数组严格表结构，改成动态规划。

去重、无效递归优化、越界结算优化：

剪枝处理

递归的几种尝试模型可改成动态规划：

1. 从左往右尝试：每个元素都是可选的（最小0次，最大1~N次），总量有限制，元素序列和总量分别对应数组下标；
2. 窗口范围尝试：尝试范围只会收缩，且左边界不能越过右边界，左右边界分别对应数组下标；
3. 样本对应尝试：两个样本大小分别对应数组下标；
4. 业务限制尝试：算出最差情况下的最大值，作为数组的一个下标。

动态规划优化：

1. 空间压缩（二维数组优化为一维数组，最省空间可尝试取行、列的较小者）：

1). 二维数组中，某个格子只依赖相邻一行的某个格子，可优化为只复用一个一维数组；

2). 二维数组中，某个格子只依赖相邻一行的某个格子和同一行的相邻格子，可优化为只复用一个一维数组+一个额外变量；

1. 迭代优化：

1). 有重复计算子问题的，可进行斜率优化（在前个已求子问题的解基础上少补多减），例如分裂数字、随机扣血量砍死怪兽等问题；

2). 无重复计算子问题的，可尝试位运算操作来优化常数时间，例如N皇后问题等问题。

1. 输入参数数组存在重复元素，可以对输入参数数组压缩去重生成2个新数组Elements和Counts，再对每个Elements[i]迭代Counts[i]次，例如有限个货币凑出目标值的最少用量等问题。
2. 状态压缩：（面试中5%以下几率）

递归输入状态数组是线性、个数有限，且每个元素只有两种状态，则可进行状态压缩，用一个int的每bit表示一个元素的状态，用int[]表示所有状态结果dp，状态结果dp的大小是2的输入数组长度的幂。

将时间复杂度由暴力O(N!)优化为了O((2^N) \* N)。

