## 37. Sudoku Solver

技巧一：善于使用/ %运算将二维参数一维化，可以节省空间，减少判断，缩减代码；

技巧二：一般凭最简单的第一思维想出来的算法都是没有什么优点的，大家都是这样想的要不断升华，反面、整体思考，搜索类的题目最重要的是优化！

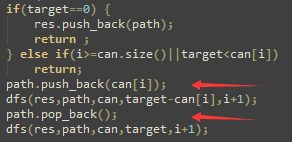
## 39. Combination Sum

## 40. Combination Sum II

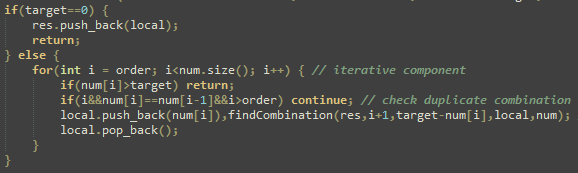
这两道题目算是回溯算法中子集树的经典变式问题。39题是集合中元素使用次数不限，40题是集合有重复元素但是求不同子集结果。一般最基本的问题是集合无重复元素且只能使用一次或者有相同元素，但是对重复不做过滤。这种变式有点像01背包问题中的多重有限无限背包问题。

一般答案为子集树搜索回溯时，有两种写法：

方法一：选择加入或不加入，递归。



方法二：依次从该位置进行加入，递归，弹出，其后元素依次操作。这种写法看上去有点像排列的味道，但是实际分析得知是O（2^n），子集树问题为T(n)=T(n-1)+T(n-2)+…T(1)，而排列树问题为T(n)=n\*T(n-1)。



同样对两种写法来说，分析的思路一样，但是程序简洁明了程度和技巧略有不同。对各种变式来说，用第二种更加方便简洁。

## 46. Permutations

## 47. Permutations II

排列问题的变式，集合有重复元素，求所有不同排列。

搜索回溯算法重要的是穷尽所有可能解，递归问题。一般有关键点在于穷尽但不重复，这种需要先清晰点的想好问题的子问题模式和保证同一位置不重复，一般可用标记，技巧一些就是利用有序性做到不重复。

## 51. N-Queens

未优化版本的回溯，纯暴力

用bit进行col的有效位置记录，缩短时间，但是用一维数组效果是一样的，此处不是重点；bit用法在多处需要判断并且需要多次传递时可节省空间和时间，只用一次时不明显

优化版本，运用数学方法映射，将行、列、45度、135度方向在O(1)的时间进行选择！

进一步写法优化，三个数组合为一个空间

搜索类的题目最重要的是优化！

## 78. Subsets

## 90. Subsets II

这道题明明跟 39. Combination Sum/40. Combination Sum II,46. Permutations/47. Permutations II是一样类型的题目，但是还卡了一会儿，做了前面四道题目，心想套路是一样的，但是卡在了对结果的思考！前面两者的记录答案的触发条件是达到结果值，而求不同子集记录结果是每次均是答案。