# 综合

## 拿到Offer的四大法宝

别做难题

是面试而不是考试

理解而不是背诵

刀要用在刀刃上

## 通用的DFS时间复杂度计算公式

O（答案个数 \* 构造每个答案的时间）

## 什么情况下使用动态规划？

满足下面三个条件之一：

1、求最大值最小值

2、判断是否可行

3、统计方案个数

则极有可能是用动态规划求解

## 什么情况下不使用动态规划？

1、求出所有**具体**的方案而非方案**个数**

2、输入数据是一个**集合**而不是**序列**

**3、暴力**算法的复杂度已经是**多项式**级别

则极不可能使用动态规划求解

# 算法总结

## 独孤九剑----总决式

最容易出卖你的，就是你的Coding Style

工程师的代码长什么样比脸长什么样重要

## 独孤九剑---破剑式

比O(n)更优的时间复杂度

几乎只能是O(logn)的二分法

## 独孤九剑---破刀式

能够用BFS解决的问题，一定不要用DFS去做！

因为用Recursion实现的DFS可能造成StackOverflow！

## 独孤九剑---破枪式

遇到二叉树的问题，就想想整棵树在该问题上的结果

和左右儿子在该问题上的结果之间的联系是什么

## 独孤九剑---破鞭式

碰到让你找所有方案的题，一定是DFS

90%的DFS题，要么是排列，要么是组合

## 独孤九剑---破索式

链表结构发生变化时

就需要Dummy Node

## 独孤九剑---破掌式

对于求2个变量如何组合的问题

可以循环其中一个变量，然后研究另外一个变量如何变化

## 独孤九剑---破箭式

BFS的主要数据结构是Queue

DFS的主要数据结构是Stack

千万不要搞反了！很体现基础知识的扎实度！

## 独孤九剑---破气式

初始化一个二维的动态规划时

就去初始化**第0行**和**第0列**

# 备注

尝试手写全域哈希、堆