**剑指offer学习笔记**

**第二章 面试需要的基础知识**

2.3数据结构

2.3.1数组

1、数组空间效率不好，时间效率好，查找O(1)，适合实现哈希表。

2、为了解决空间效率不高的问题：STL模板库有vector。Vector：当数目超过数组容量时，开辟一个二倍的空间，原数组复制过去，再把原来的释放，对时间有不利影响。

3、C/C++没有记录数组的大小，程序员要自己判断边界问题。

4、数组作为参数传递时，会自动蜕化为同类型的指针。

习题：

面试题3：二维数组中的查找

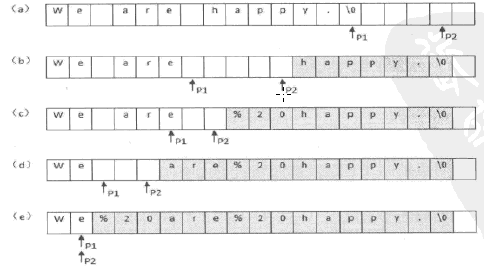
2.3.2字符串

1、为了节省内存，C/C++把常量字符串放到一个单独的内存区域。当几个指针赋给相同的常量字符串时，它们实际上会指向相同的内存地址。初始化数组的时候却不一样。

习题：

面试题4：空格替换

时间复杂度O(n)



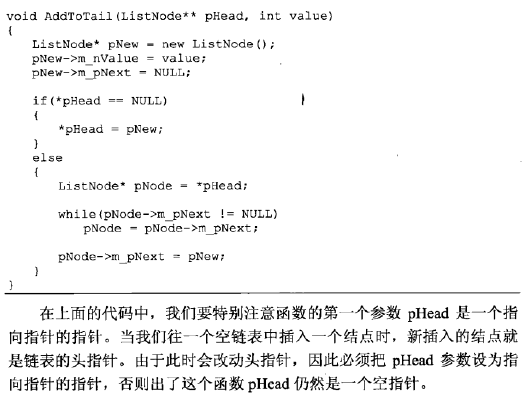
本题考点：

1、时间复杂度的分析

2、意识到潜在的问题，字符覆盖后，字符串的长度还够用吗

2.3.3 链表

1、



习题：

面试题5：从尾到头打印链表（典型的后进先出问题）

可选方法:

1、改变指针

2、栈

3、递归

2.3.4树

1、堆（通常解决最大值最小值问题）和红黑树（根到叶的最长路径长度不超过最小的两倍）

习题：

面试题6：重建二叉树（同王道机试P53）

1、里边有两个值得关注的地方是异常处理，解决输入不合法的情况。（前序遍历和中序遍历不匹配）

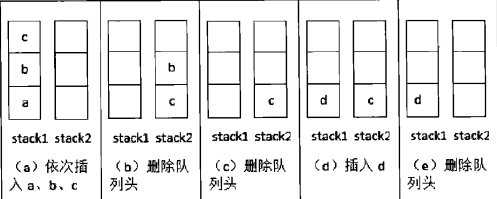
2.3.5栈和队列

1、堆（通常解决最大值最小值问题）和红黑树（根到叶的最长路径长度不超过最小的两倍）

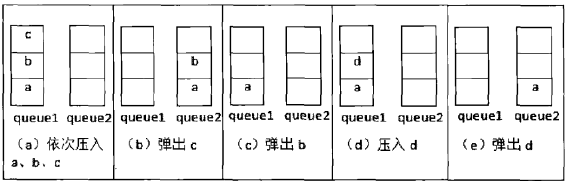
2、应用：比如线程

习题：

面试题7：两个栈实现队列



相关：两个队列实现栈



2.4 算法和数据帧操作

2.4.1 查找和排序

1、在排序的数组中查找一个数字或者某个数字出现的次数，都可以尝试二分查找

2、面试官常会问各种排序的优劣

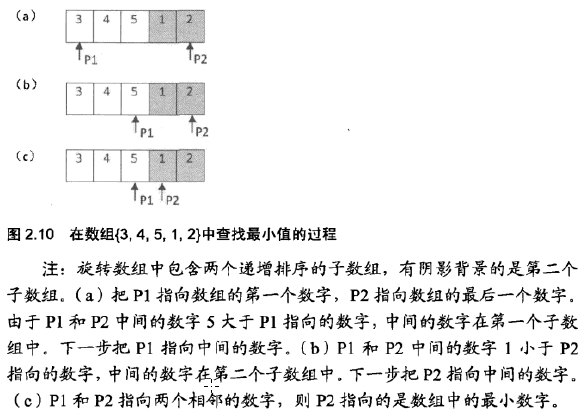
3、在被要求是实现排序算法的时候，一定问清楚比如数字的范围，时间复杂度的要求等条件，再选择算法。

习题：

面试题8：旋转数组的最小数字

本题给出的数组在一定范围上是有序的，可以试着使用二分查找的方法。

特殊情况：0旋转。（其实就是初始值的设立）



2.4.2 递归和循环

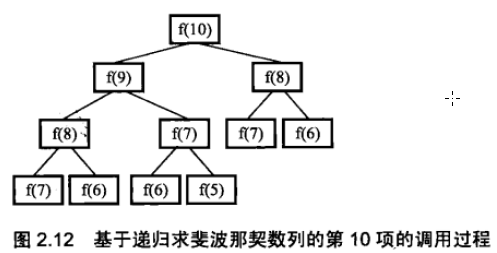
1、能用循环尽量不用递归

递归的坏处：

1、时间和空间消耗大

2、调用栈可能溢出

3、可能很多计算是重复的：

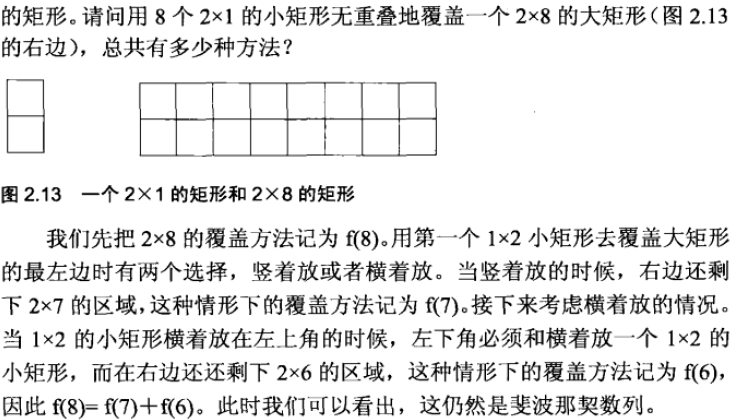


习题：

面试题9：斐波那契数列/青蛙跳台/青蛙跳台扩展/矩形覆盖

青蛙跳台扩展：f(n) = f(n-1) + f(n-2) + f(n-3) + ... + f(n-(n-1)) + f(n-n) =2\*f(n-1)

矩形覆盖：



2.4.3 位运算

习题：

面试题10：二进制中1的个数

1、 考虑正负：(负数)0x80000000<<1=0xC0000000而不是0x40000000，0xC0000000一直移位，最终会变成0xFFFFFFFF

2、惊喜的做法：

一个整数减去1，再与原来相乘，n\*(n-1)相当于将二进制最后一个1变成了0；

利用这个原理，可以判断一个数是不是二的整数次方

3、利用异或结果中1的位数，可以两个整数的二进制位最少改变多少可以相同。

4、n\*(n-1)这个原理很重要！！

**第三章 高质量的代码**

3.2 代码的完整性

1、例如：将字符串转化为整数。考虑：

1、0，整数，负数

2、long long是否可以接收这个数，不可以怎么办

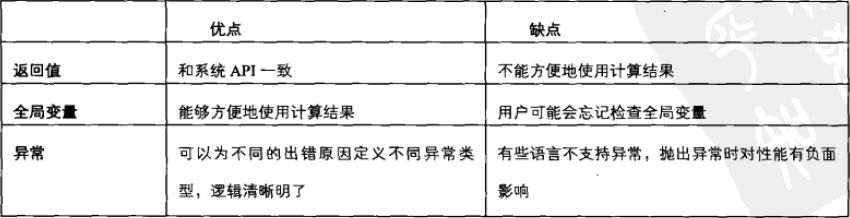
3、负面测试用例，1a2b3cz

2、例如：对于递归和循环，边界终止是否正确

3、将可能的变化考虑进去，在需求变化发生的时候，才可能尽量减少改动的风险。

4、机器内表示float和double时都有误差，不能使用==判断相互关系，只能通过看二者之间的差值是否很小，比如小于0.0000001.

5、错误处理的方法：



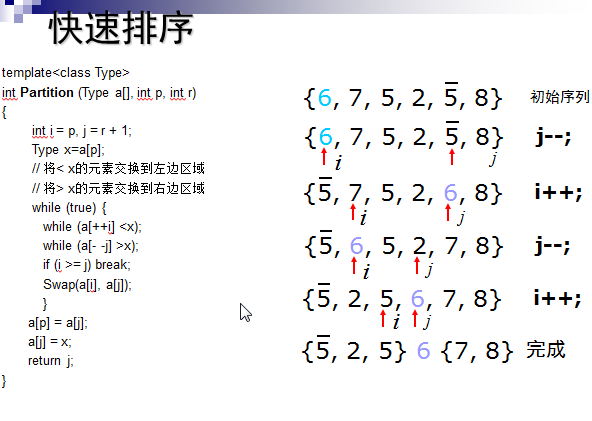
习题：

面试题11：数值的整次方

指数是1，负数，0的时候怎么办

面试题14：调整顺序使奇数位于偶数前面

1、一种类似于快排的不稳定解法



2、函数的解耦：将有可能不断改变的地方，另外写成一个函数

3.4代码的鲁棒性

1、防御性编程，如：试图打开文件时，文件不存在，我们可以提示用户检查文件名和路径；服务器连接不上，试图连接备用的；

2、如果输入的指针空怎办，如果字符串空怎么办。

习题：

面试题15：链表中倒数第k个节点

1、空指针怎么办

2、节点数小于k怎么办

3、for(i=1;i<k;i++)也不用for(i=0;i<（unsigned int）k-1;i++)，k=0时会死循环

相关问题：（当一个指针不能解决链表的所有问题时，p，q速度不同或者一个先走几步）

1、链表的中间节点;：p，q，一个一次走两步，另一个走一步

2、单链表是否为环形：p，q，一个一次走两步，另一个走一步，看慢的可否追上快的

面试题 16：反转链表

做之前考虑要保存什么

面试题 17：合并两个排序链表

哇！可以递归

面试题 18：树的子结构

树的问题每一次涉及指针都判断是否为NULL

**第四章 解决面试题的思路**

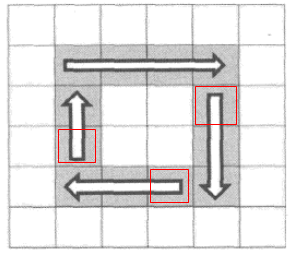
4.2画图让抽象问题形象化

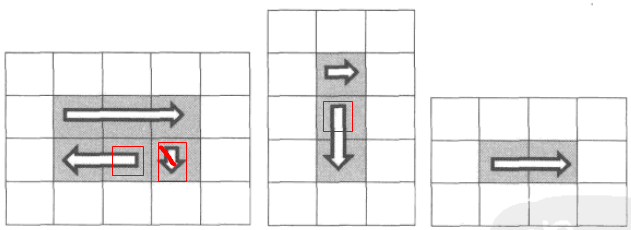
习题：

面试题19：二叉树的镜像

面试题20：逆时针打印矩阵

搞清楚循环要不要进来，进来几下，起始点的限定条件是什么，起始点该不该有





4.3 让抽象问题具体化

习题：

面试题 21：包含min函数的栈

面试题 22：栈的压入和弹出序列

面试题 23：从上到下打印二叉树

面试题 24：二叉搜索树的后序遍历

面试题 25：二叉树中和为某一值的路径

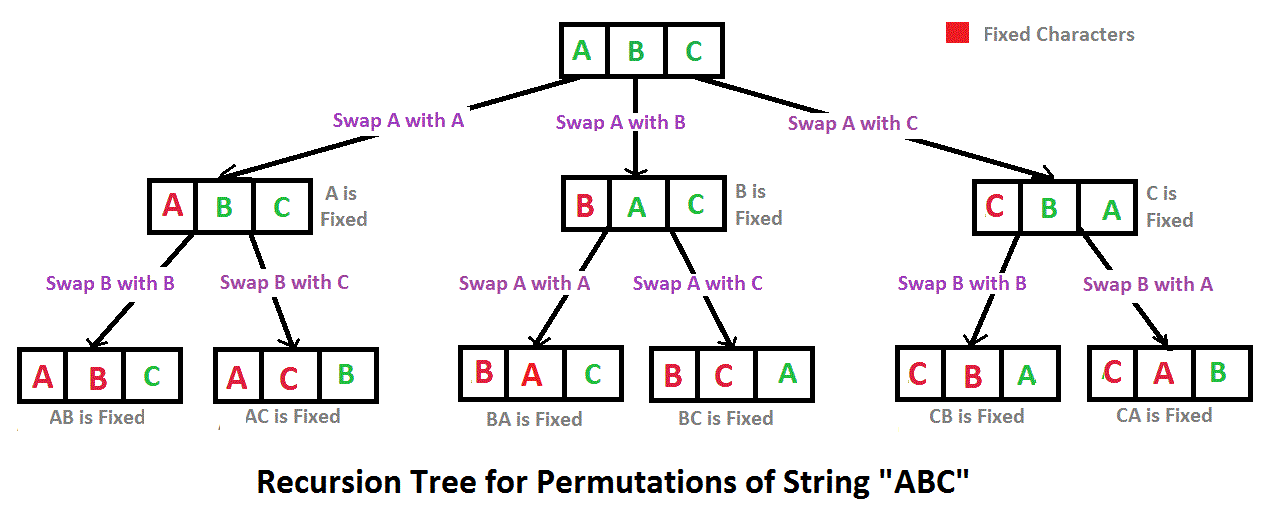
4.4分解让复杂问题简单化

习题：

面试题26：复杂链表的复制 NULL没有next！！！

面试题27： 二叉搜索树与双向链表/////中序遍历的改版

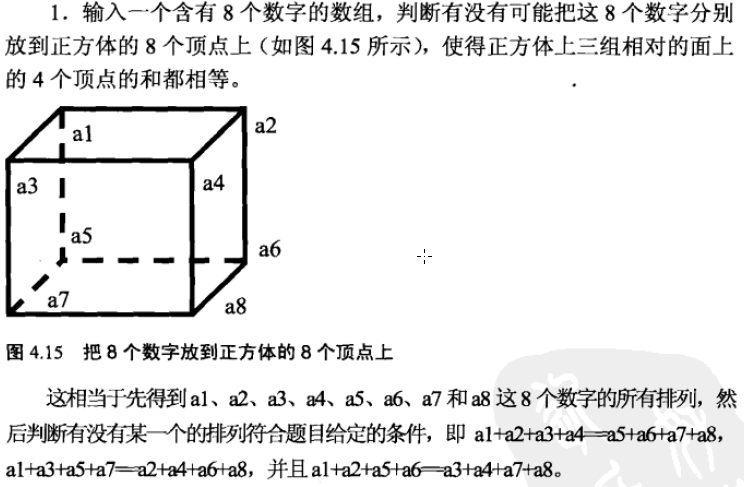
面试题28： 字符串的排列



问题扩展：

1、所有字符的组合

2、八皇后

3、

4.5 本章小结

遇到复杂问题可以画图，举例子，分解

**第五章 优化时间和空间效率**

5.1 面试官谈效率

1、普通的应用是拿空间换时间，因为用户更加关心速度，而且一般有足够的存储空间。但是对于嵌入式设备，很多时候是拿时间换空间。

5.2 时间效率

1、C/C++里要严惩养成使用引用和指针传递复杂参数的习惯，如果是值传递会有一次赋值费时。

2、字符串string的拼接，尽量少用+，因为会产生很多临时实例，会费时，尽量使用append

3、循环和递归的时间效率可能不一样，递归时小问题可能会重叠，虽然简洁，但是可能会重复计算。

动态规划的时候，喜欢使用数组保存中间结果。

4、查找：顺序查找O(N)，对于排序数组二分查找O(logN)，哈希O(1)

习题：

面试题29：数组中出现次数超过一半的数字(cnt\*2>len)

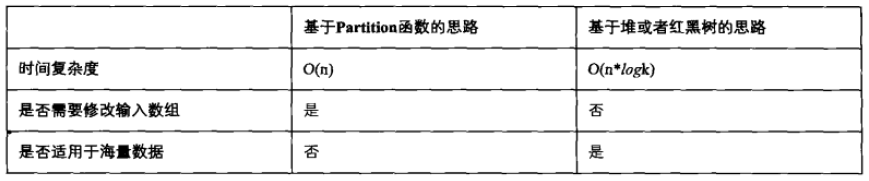
1、输入数组是否无效

2、是否可以修改数组

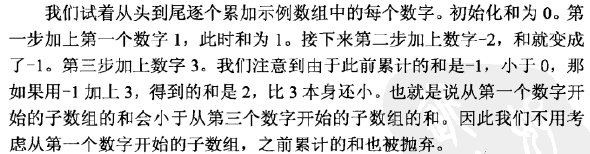
面试题30：最小的k个数

1、大顶堆 priority\_queue

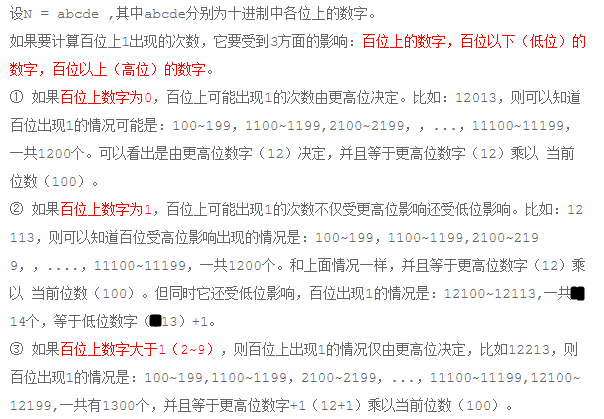
2、红黑树 multiset set（不允许有重复数字）（适合于海量数据，不改变原数组）



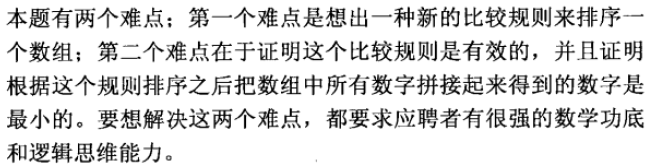
面试题 31：连续子数组的最大和///动态规划



面试题 32： 整数中1出现的次数（O(logN)）



面试题 33: 把数组排成最小的数



5.3时间效率和空间效率的平衡

习题：

面试题 34：丑数

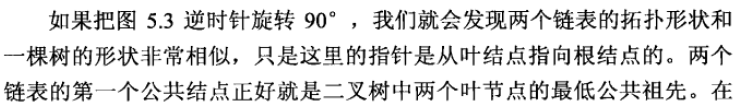
面试题 35：第一次只出现一次的字符（哈希的应用）

面试题 36：数组中的逆序对（二路归并的变形）

面试题 37：两个链表的第一个公共结点

1、使用栈

2、大的比小的长k，大的先走k，再一个个地比较



5.4 本章小结

嵌入式系统对内存限制大，在空间换时间的时候要多加考虑

**第六章 面试中的各项能力**

6.2 沟通能力和学习能力

1、多提问，以明确需求

举例：两棵树的最低公共祖先

是二叉树吗 是排序的二叉树：遍历的变形

不是二叉树，但有指向父亲的指针：链表的第一个公共结点问题不是不是二叉树，没有指向父亲的指针：利用栈

6.3知识迁移能力

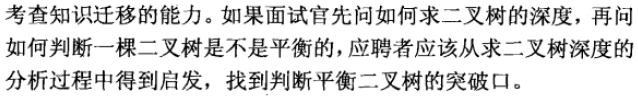
习题：

面试题38：数字在排序数组中出现的次数

1、二分再向左向右O(logN)+O(N)

2、改造二分，直接找到第一个和最后一个k，O(logN)

面试题39：二叉树的深度（遍历的改造）



面试题40：数组中只出现一次的数字

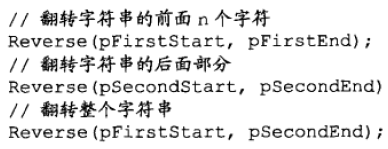
1、题目为什么要强调有一个数字出现一次，其他的出现两次？我们想到了异或运算的性质：任何一个数字异或它自己都等于0 。也就是说，如果我们从头到尾依次异或数组中的每一个数字，那么最终的结果刚好是那个只出现一次的数字，因为那些出现两次的 数字全部在异或中抵消掉了。

2、 我们还是从头到尾依次异或数组中的每一个数字，那么最终得到的结果就是两个只出现一次的数字的异或结果。因为其它数字都出现了两次，在异或中全部抵消掉了。由于这两个数字肯定不一样，那么这个异或结果肯定不为0 ，也就是说在这个结果数字的二进制表示中至少就有一位为1 。我们在结果数字中找到第一个为1 的位的位置，记为第N 位。现在我们以第N 位是不是1 为标准把原数组中的数字分成两个子数组，第一个子数组中每个数字的第N 位都为1 ，而第二个子数组的每个数字的第N 位都为0 。现在我们已经把原数组分成了两个子数组，每个子数组都包含一个只出现一次的数字，而其它数字都出现了两次。因此到此为止，所有的问题我们都已经解决。

面试题41：和为S的两个数字VS和为S的正数序列



面试题42：翻转单词顺序VS左旋转字符串



6.4 抽象建模能力

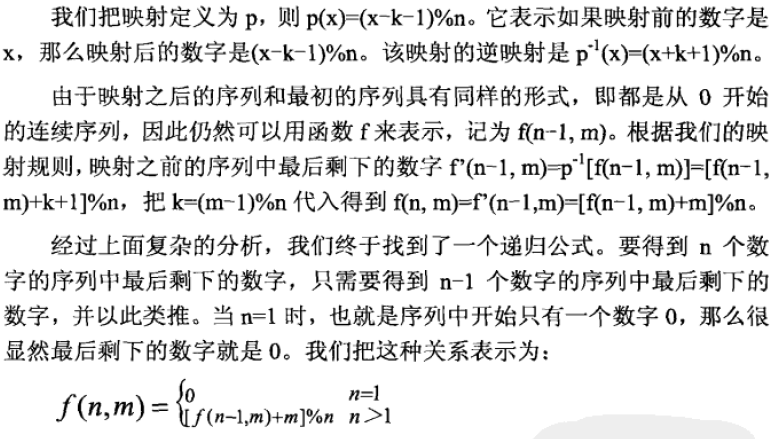
习题：

面试题44：扑克牌的顺子

面试题45：圆圈中最后剩下的数字

1、numbers.erase(指针);

2、数学推导



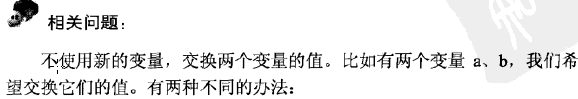
6.5发散思维能力

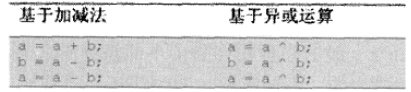
习题：

面试题46：1+2+……+n

面试题47：不用加减乘除做加法

1、^

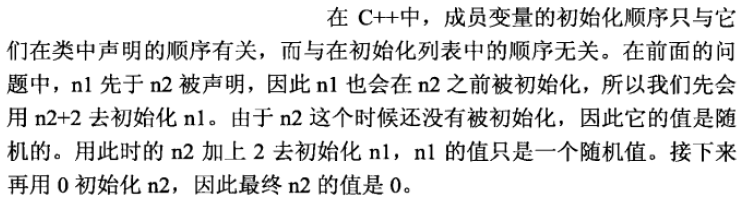


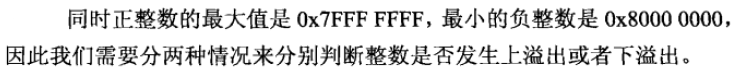


**第七章 两个面试案例**

7.1案例一

面试题49：把字符串转换成整数





**第八章 英文版新增面试题**

8.1 数组

习题：

面试题51：数组中重复的数字

由于题目的特殊性，不需用hash，不需用sort，从头到尾扫描，把每一个数字换到它该在的地方，如果发现该在的地方已经是他了，返回有重复。

面试题52：构建乘积数组

由于不可以使用除法，使用两次循环正着乘一遍，再反着来一遍，就可以。

8.2 字符串

习题：

面试题53：正则表达式匹配

对于非确定有穷状态自动机采取试探的方式

面试题 54：表示数字的字符串

关键在于确定字符串的匹配模式

面试题55：字符串中第一个不重复的字符

Hash

8.3 链表

习题：

面试题56：链表中环的入口结点

1、一个走快一点，一个走免慢一点，采取不同的步伐，找到环的存在

2、计算环的大小n

3、p1，p2一个先走n步，然后一起走，直到p1==p2，就是环的入口

面试题57：删除链表中重复的节点

p->val==pNext->val

8.4 树

习题：

面试题58：二叉树的下一个结点

1、右子树的最左节点

2、一直向上找到是它父节点的左孩子的节点

面试题59：对称二叉树

isSymmetrical(left->left,right->right)&&isSymmetrical(left->right,right->left);

面试题60：把二叉树打印成多行

设置一个lastnode，，每当lastnode换行

面试题61：按之字形打印二叉树

利用两个栈

面试题63：二叉搜索树的第k个节点

中序遍历到第k个节点为止

面试题64：数据流中的中位数

下一半最大堆，上一半最小堆

8.5 栈和队列

习题：

面试题65：滑动窗口的最大值

利用双端队列，每次保存可能成为最大值的点，每次也判断，不和规则的舍去，确保双端队列的头保存的是每一次的最大

8.6 回溯法

习题：

面试题66：矩阵中的路径

面试题67：机器人的运动范围