Homework2

**1.Two Sum**

\*构建新的数组

**int**[] nums = **new** **int**[]{1,3,4};

\*返回无效值

在末尾加上

**throw** **new** IllegalArgumentException("No two sum solution");

**160.Intersection of Two Linked Lists**

\*思路：

（自己一开始没有考虑到）

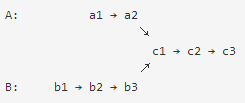
从intersection节点开始两个list共享长度

\*Solutions上的奇妙思路：

\_\_\_\_\_\_\_\_\_\_a\_\_\_\_\_\_\_\_ \_\_\_b\_\_\_

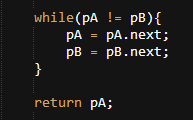
\_\_\_b\_\_\_ \_\_\_\_\_\_\_\_\_\_a\_\_\_\_\_\_\_\_

a的末尾一段和b的末尾一段自然而然地对接起来



\*写程序的过程中没有考虑到的corner case

两条链子同样长度，没有intersection，应该返回null。运气好程序自带了这个功能。



\*疑问

给定ListNode p1, ListNode p2.

p1==p2 的充要条件是什么？

是不是p1.val==p2.val && P1.next=p2.next

**349. Intersection of Two Arrays**

\*最初没有理解题意，以为要求“重合的一段array，对位置敏感”。

实际只要求“有重合的几个点，对位置不敏感”

**350. Intersection of Two Arrays II**

\*最初没有理解题意，以为要求“重合的一段array，对位置敏感”。

实际只要求“有重合的几个点，对位置不敏感，重合点数值可重复”

有点类似于two pointers。有时滑动一个，有时滑动俩

**190. Reverse Bits**

方法一：Bruteforce

\* >> 与>>> 区别

>>是算术右移， 在左端补k个最高位有效值

>>>是逻辑右移，在左端补k个0

本题要用逻辑右移

\*需要整理一下bit manipulation中的& || ^ xor 用法

注意&& 与& 区别

注意|| 与| 区别

方法二：

Swap 16位，8位，4位，2位

！！！明天起床后自己写一遍方法二

**20. Valid Parentheses**

**35. Search Insert Position**

思路与367一样。

**367. Valid Perfect Square**

\*l,r,mid都要用long

如果l,r,mid用int，会出现mid\*mid 溢出的问题

=>e.g. 

小知识：int 32位、long 64位

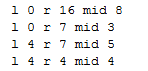
\*思路

Mid=(l+r)/2

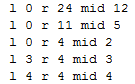
=>Case1: l = mid+1;

=>case2: r = mid-1;

e.g.Input = 16



e.g.Input = 24



？此处还要总结一下规律