## Database

1. ***Second Highest Salary***

limit 别名 asc（升序，从小到大） desc（降序，从大到小）

group by 的效率貌似比 distinct 高点

***175. Combine Two Tables***

左外连接，还有别名，至于那个as，你说到底要不要写呢？写起来好烦啊。

***181. Employees Earning More Than Their Managers***

自连接

所谓连接，就是将右表的数据与左表拼凑到一起，on后跟的就是拼凑条件。这样想不知道对不对啊。

1. ***Duplicate Emails***

自连接，同时注意distinct

group by & having，where后不能跟函数。

1. ***Customers Who Never Order***

左外连接，且不要重合的数据。Null不能用 = 只能用 is（唉，真他妈烦）

MySQL没有全外连接

1. ***Rising Temperature***

时间函数，on里头可以用加与减。（乘除能不能用？）

1. ***Delete Duplicate Emails***

Delete语句内也可以使用连接。

MySQL禁止在from子句中指定被更新的表，也就是说，IMG_256是错的：

delete from *person* where *id* not in (select *id* from *person* group by *email*);

// 坑爹呢这是。。。

1. ***Classes More Than 5 Students***

SELECT

c.class

FROM

courses c

GROUP BY

c.class

HAVING

count(DISTINCT c.student) > 4

没想到吧，distinct还可以用在这种地方。记住，子查询的效率一定是低的，没办法的情况下再用。

**///////////////////////////////////////////////////////////////////////////////////**

## Algorithms

1. Number of Boomerangs

暴力穷举，将每一种组合方式都列出来，然后在比较点与点之间的长度（计算平方），就结而论，失败了，Time Limit Exceeded。

我另一个想法是找出这三个点的一元一次函数的中垂线函数，并没有实践，但就算成功了，也快不到哪去，n3摆在那。

通过HashMap；嵌套循环；计算点与点之间的距离；比对（懒得组织语言了）

但光距离相同是不行的，是没有什么软用的，因为一个不好就统计成四个点了，这是不符合题目要求的。

所以hashmap不能放在循环外。

496. Next Greater Element I

两个整数数组findNums，nums，均无重复数据。findNums是nums的子集，目的在于从nums中按序找出比findNums中元素大的那个元素。。。。怎么越说越玄乎了

**Input: nums1 = [4,1,2], nums2 = [1,3,4,2].**

**Output: [-1,3,-1]**

**Explanation:**

**For number 4 in the first array, you cannot find the next greater number for it in the second array, so output -1.**

**For number 1 in the first array, the next greater number for it in the second array is 3.**

**For number 2 in the first array, there is no next greater number for it in the second array, so output -1.**

还是上例子吧。

起初，因为题目叫next greater element，所以在定位元素位置后，我只是单纯的比较了后一个元素，小于赋值，其它情况赋-1。结果这东西它告诉我是要一直向后找，直到最后一个元素的。

**Input: nums1 = [1,3,5,2,4] ，nums = [6,5,4,3,2,1,7]**

**Output: [7,7,7,7,7]**

啊，人生。

1. Array Partition I

这道题目写的相当失败，因为一直没理解题目是什么意思。

Given an array of **2n** integers, your task is to group these integers into **n** pairs of integer, say (a1, b1), (a2, b2), ..., (an, bn) which makes sum of min(ai, bi) for all i from 1 to n as large as possible.

我现在不能很流畅的表述出来，给定一个长度为2n的数组，将其分为n个对数，问这n个对数中偏小的数之和最大为多少。

// 还是很绕。。。

唉，也不是自己想出来的

**///////////////////////////////////////////////////////////////////////////////////**

## 番外

***人生的第二次面试，那个人是CTO啊***

给定一随机矩阵，将其顺时针翻转90度。转换一下就是翻转二维数组，假设转换前的数组是before，转换后的数组是after。那么，before的行是after的列，before的列是after的行；before最后一行第一列的数据是after第一行第一列的数据，一次类推。剩下的是如何给after赋值，有两种选择，从行开始或从列开始。

这个解法是从行开始赋值的，因此，数据从before的第一列，最后一行依次往上。

**int**[][] after = **new** **int**[before[0].length][before.length];

**for**(**int** i = 0; i < before[0].length; i++){

**for**(**int** j = 0; j < before.length; j++){

after[i][j] = before[before.length - 1 - j][i];

}

}

这个解法是从列开始赋值的，因此，数据从before的最后一行，第一列依次往后。

**for**(**int** i = 0; i < before.length; i++){

**for**(**int** j = 0; j < before[0].length; j++){

after[j][i] = before[before.length - 1 - i][j];

}

}

## 数据结构与算法分析

* 二叉树

说是要总结，可真打开word，心里又很纠结，因为不知道该说些什么。提起二叉树，我只能联想到递归。

if（root == null）{

return null; / return 0; / return XXX;

}

method(root.left);

method(root.right);

// TODO... ...

凡这种写法都是深度遍历优先，递归至左子树的最末端。

广度遍历优先，需要借助另一种数据结构来完成（可能不需要，但目前我不会）。

队列。

队列的特点是先进先出，后进后出。那么新建一个队列，把二叉树的根节点（root）插入（offer）队列，然后从首端弹出（peek）。弹出时在检测该节点是否存在子节点，有的话继续插入队列，继续弹出。所以说这是一个while循环，哈哈哈哈，终于不是递归了。