**1.LRU cache.** 给一个array, 给一个cache max size, 输出miss count. 什么时候hit, 什么时候miss的情况写好就好了。

example： size = 4， input array 【1，2，3，4，5，4，1】

1 miss 2 miss 3 miss 4 miss 5 miss 替换 1

4 hit 把4提前到第一位 1 miss 替换 2

**2.给一个数组 {0,0,1,1,0,1}** 和 一个数字n，flip这个数组n次，flip规则：左右不同的设为1，相同设为0。就是XOR操作。数组头尾（index = -1 or arr.length）可以当作是0，之前也有哥们遇到此题http://www.1point3acres.com/bbs/thread-138007-1-1.html

一个八个元素的int array，值为0或者1，每天晚上变一次，怎么变呢。。看两边的元素，如果两边一样，变0，不一样变1。最左边和最右边的左边和右边就算做0。举个栗子：00111001，第二天变为01101110（应该没错=。=） 两个参数，一是初始状态，二是总共几晚上

ps：两端cell两边假设const = 0；

小example：

day0: (0)0,1,1,1(0)

day1: (0)1,1,0,1(0)

day2: (0)1,1,0,0(0)

**3.robin wait time。**

大概是这样，给一个int[] arrival time, int[] Execution time, int q. 例子： 【0，1，4】 【5，2，3】 q=3. 输出的是average wait time 2.3333333

**4.给定一个BST的root，找从root到leaves的权值最小的路径。不**要求返回路径，只要求返回路径上节点值的sum。要求这个sum最小。用divide and conquer就行。注意这题有null node，以及它的null left和null right。。。这结构见了鬼

5.第二个是给出一个已经排序的循环链表，**往里面插入一个节点**。http://www.geeksforgeeks.org/sorted-insert-for-circular-linked-list/ 实际的题有一点不一样的是它每次给你的HEAD不一定是链表最小那个。题目定义了一个CNode，其实跟ListNode一样，有val，有指针，list首尾相连而已，list里的node升序排列，题目也不难，不过有两点注意一下：一个是给你的CNode start不一定是最小值的CNode，所以要先找到最小的点；

6.第二道题我遇到的是**SHORTEST JOB FIRST**。 给你的INPUT是两个ARRAY，一个是每个JOB的START TIME，一个是每个JOB的EXECUTION TIME。

叫你求AVERAGE WAITING TIME。 学完OS课忘了，或者没学过的，都不用急， 它的描述里讲的很清楚怎么求，认识读几遍就行了。

7. 一道**rotate m\*n matrix**，没有很难，跟leetcode rotate image差不多，只不过是m\*n而已,可以参见leetcode https://leetcode.com/problems/rotate-image/， 不过不一样的是1.不是正方形的 2.给一个flag参数用来决定是往左还是往右。如果之前的算法不是in space的话应该没有啥差别。

**8. Greates Common Divisor；**输入是一个int数组，输出是所有int类型的最大公约数。

我的思路是，一个数组的最大公约数一定是某两个数最大公约数和其他数的最大公约数

所以求出来前两个的最大公约数，再拿到这个数字和之后所有的数字求最大公约数就是结果

**9. find path in 2D matrix，**输入一个2D int array,可能是三种情况，0，1，9。 0表示可以走， 1表示障碍不能通过，9表示最终结果。出生点在（0，0），返回true/false 表示能不能从（0，0）走到有9的位置上。并没有明说9是否唯一。