1. 整除

（div.cpp）

限制：1S　　128MB

【问题描述】

给定 N，请求出既不能被 2 整除也不能被 3 整除的第 N 个正整数。

【输入】（div.in）

输入为 N。

【输出】（div.out）

输出为一个正整数。如既不能被 2 整除也不能被 3 整除的第 N 个正整数；

(提示：请使用 long long 类型，输出结果保证不超出 long long 类型范围)。

【输入输出样例】

样例 1	样例 2
输入 5 输出 13	输入 100000 输出 299999

【样例说明】

样例 1：不能被 2 和 3 整除的前 5 个数为：1、5、7、11、13，第 5 个数为 13；

【数据范围】

对于30%的数据，保证 N<=10^5。

对于100%的数据，保证 N<=10^9。

2. 填积木

（block.cpp）

限制：1S　　128MB

【问题描述】

有一块长为 m,宽为 n，高为 h 的魔幻空间，需要你用长宽高都是 a 的正方体积木填满整个空间（积木可以超出这块魔幻空间的范围)，你不能把积木打碎，请问你至少需要多少块积木才能把魔幻空间填满。

【输入】（block.in）

输入为 4 个整数 n, m, h 和 a，分别表示魔幻空间的长宽高和正方体积木的长宽高。

【输出】（block.out）

输出为一个正整数表示至少要多少块积木

(提示：请使用 long long 类型，输出结果保证不超出 long long 类型范围)。

【输入输出样例】

样例 1	样例 2	样例 3	样例 4
输入 5 1 1 2 输出 3	输入 10 10 1 3 输出 16	输入 4 5 6 2 输出 18	输入 1234567 7654321 3333333 213 输出 3260223524800

【样例说明】

样例 1：长为 5,宽和高都为 1 的魔幻空间，积木的边长为 2，用 3 块积木才能填满；

【数据范围】

有 20%的数据，保证 1≤n,m,h,a≤100。

有 10%的数据，保证 1≤n,m,h≤10^6,a=1。

有 20%的数据，保证 1≤n,a≤10^9,m=h=1。

有 20%的数据，保证 1≤n,m,a≤10^9,h=1。

对于 100%的数据，保证 1≤n,m,h,a≤10^9。

3. 闰年

（year.cpp）

限制：1S　　128MB

【问题描述】

能被 4 整除但不能被 100 整除的年份是闰年；能被 400 整除的年份也是闰年。比如 2024 年是闰年，2022 年不是闰年；2000 年是闰年，2100 年不是闰年； 给出正整数 a 和 b,求第 a 年到第 b 年之间有多少个闰年(包括 a 和b)。

【输入】（year.in）

输为两个用空格隔开的整数 a 和 b(a<=b)。

【输出】（year.out）

输出为一个数，表示有多少个闰年。(提示：请使用 long long 类型)

【输入输出样例】

样例 1	样例 2	样例 3	样例 4
输入 2000 2024 输出 7	输入 1 1000 输出 242	输入 1 1000000 输出 242500	输入 1 100000000000 输出 24250000000 输入说明：1 后面有 11 个 0）

【数据范围】

对于 30%的数据，保证 1<=a,b<=100;

对于 50%的数据，保证 1<=a,b<=10^6;

对于 100%的数据，保证 1<=a,b<=10^{12}。

4. 发牌

（card.cpp）

限制：1S　　128MB

【问题描述】

小雨同学在玩发牌的游戏，她有 N 张牌，第一张牌的数字是 1，第二张的数字是 2，第三张的数字是 3，...，第十三张的数字是 13，第十四张的数字是 1，...，以此类推，第 n 张牌的数字是(n-1)%13+1;发牌的方式是“藏一发一”，把第 1 张放到最后，发第 2 张，把第 3 张放到最后，发第 4 张，把第 5 张放到最后，发第 6 张，...，一直这样发下去，直到剩下一张牌为止，问剩下的最后一张牌的数字是多少？

【输入】 （card.in）

输入为一个数 N，表示牌的数量。

【输出】 （card.out）

输出最后一张牌的数字。

【输入输出样例】

样例 1	样例 2	样例 3	样例 4	样例 5
输入 13 输出 11	输入 3 输出 3	输入 1000 输出 2	输入 1000000 输出 7	输入 10000000000 输出 10

【样例说明】

样例 1 解释：发牌的顺序为：2,4,6,8,10,12,1,5,9,13,7,3,11。

样例 2 解释：发牌的过程为：把第 1 放到最后，发 2，把 3 放到最后，发 1， 还剩一张 3；

【数据范围】

对于 20%的数据，保证 1<=N <=13。

对于 40%的数据，保证 1<=N <=1000。

对于 100%的数据，保证 1<=N <=10^{15}。

5. 二叉查找树

（bst.cpp）

限制：1S　　128MB

【问题描述】

相信大家对二叉查找树都很熟悉了，现在给你 N 个整数的序列，每个整数都在区间[1,N]内，且不重复。现在要你按照给定序列的顺序，建立一个二叉查找树，把第一个整数作为根，然后依次插入后面的整数。

每个结点X的插入过程其实就是模拟下面的 **insert(X, root)**过程：

```
insert( number X, node N )
{
    increase the counter  C  by  1  //每次进来都会使C加1
    if X is less than the number in node N //如果X小于结点N的值
    {
        if  N has no left child //N没有左孩子
            create a new node with the number X and set it to be the left child of node
            N //新建一个节点，把X作为N左孩子
        else  insert(X, left child of node N)//递归插入到N左子树
    }
    else (X is greater than the number in node N) //如果X大于结点N的值
    {
        if  N has no right child  //N没有右孩子
            create a new node with the number X and set it to be the right child of
            node N//新建一个节点，把X作为N右孩子
        else
            insert(X, right child of node N) //递归插入到N右子树
    }
}
```

你的任务是：每次把序列的一个整数插入到二叉查找数后，到目前为止计数累加器C的值是多少？请把它输出。

注意：**第一次插入根，计数器C的值是0**，你可以理解为插入根是不执行insert()操作的，其后每插入一个结点，C都累加，也就是每次进入过程 **insert(number X, node N)，都会执行increase the counter C by 1，使得C不断增大。**

【输入】（bst.in）

第一行一个整数：N, 表示序列有多少个整数。

接下来有 N 行，每行一个整数 X, X 在区间[1,N]内，且不重复，这 N 个整数就组成了一个有序的序列。

【输出】（bst.out）

N 行，每行一个整数，第一行表示你把序列的第一个数插入到二叉查找树后，当前计数器 C 的值是多少。

【输入输出样例】

输入：	输出：
8 3 5 1 6 8 7 2 4	0 1 2 4 7 11 13 15

【样例说明】

（ 第一次插入根1 ， 不执行过程 insert(number X, node N)，所以C是0，第二次插入2, C加1，所以插入2完毕后，输出C的值是1，第三次插入3，依次执行了insert(3, 1) 、insert(3,2)，所以C 加2，变成3； 第四次插入4，依 次 执 行 了 insert(4, 1) 、 insert(4,2) 、insert(4,3)， 所以C 加3，变成6。

【数据范围】

20%的数据：1 <= N <= 1000

100%的数据：1 <= N <=300000