## 2028 Lowest Common Multiple Plus

### 一、题目

#### **问题描述**：

求n个数的最小公倍数。

#### **输入数据**：

输入包含多个测试实例，每个测试实例的开始是一个正整数n，然后是n个正整数。

#### **输出数据**：

为每组测试数据输出它们的最小公倍数，每个测试实例的输出占一行。你可以假设最后的输出是一个32位的整数。

#### 样例输入：

2 4 6  
3 2 5 7

#### 样例输出：

12  
70

#### **题目来源**：

hdu 2028 http://acm.hdu.edu.cn/showproblem.php?pid=2028

### 二、题解

#### **解题思路**：

2个数的最小公倍数是其乘积除以最大公约数，那么n个数的思路就是先解出前两个数的最小公倍数，然后用该数和第三个数求其二的最小公倍数…..

#### **参考程序**

#include<stdio.h>  
int a[10000];  
  
int gdc(int a, int b)  
{  
 return a == 0 ? b : gdc(b % a, a); //辗转相除求最小公约  
}  
  
  
int main()  
{  
 int n;  
 while (scanf("%d", &n) != EOF)  
 {  
 int k, l = 1;  
 int t;  
 for (int i = 0; i < n; i++)  
 {  
 scanf("%d", &a[i]);  
 }  
 for(int i=0;i<n-1;i++) //将数组从小到大冒泡排序，实现gdc函数中a<b  
 {  
 for(int j=0;j<n-1-i;j++)  
 {  
 if(a[j]>a[j+1])  
 {  
 int r=a[j];  
 a[j]=a[j+1];  
 a[j+1]=r;  
 }  
 }  
 }  
 k = gdc(a[0], a[1]);  
 l = a[0] \* a[1];   
 t = l / k; //前两个的最小公倍数  
 for (int i = 2; i < n; i++) //循环实现n个数的最小公倍数  
 {  
 k = gdc(t, a[i]);  
 t = t / k \* a[i];  
 }  
 printf("%d\n", t);  
 }  
 return 0;  
}

#### 复杂度分析

无

#### 实现技巧：

懂得简单的递归调用(辗转相除)