**Problem A: 编写函数：判断闰年 (Append Code)**

Time Limit: 1 Sec  Memory Limit: 2 MB  
Submit: 12784  Solved: 7029  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=0&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1465)]

**Description**

输入一个正整数的年份，判断是否为闰年。

-----------------------------------------------------------------------------

编写一个函数用于判断闰年。

用C语言实现：append.c中函数原型为

int is\_leap\_year(int year);

功能：若参数year是闰年返回1，否则返回0。

用C++实现：append.cc中函数原型为

bool isLeapYear(int year);

功能：若参数year是闰年返回true，否则返回false。

函数的调用格式见“Append Code”。

**Input**

输入只有一行，为一个10000以内的正整数。

**Output**

输出为一行。  
若输入为闰年偶数则输出“Yes”，否则输出“No”。

**Sample Input**

2010

**Sample Output**

No

**HINT**

**Problem B: 编写函数：求最大公约数gcd()和最小公倍数lcm() (Append Code)**

Time Limit: 1 Sec  Memory Limit: 2 MB  
Submit: 15413  Solved: 6351  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=1&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1228)]

**Description**

辗转相除法，也称欧几里得算法，是求最大公约数的算法。辗转相除法首次出现于欧几里得的《几何原本》（第VII卷，命题i和ii）中，而在中国则可以追溯至东汉出现的《九章算术》。  
两个整数的最大公约数(亦称公约数)是能够同时整除它们的最大的正整数。辗转相除法基于如下原理：两个整数的最大公约数等于其中较小的数和两数的差的最大公约数。例如，252和105的最大公约数是21（252 = 21 × 12；105 = 21 × 5）；因为252 − 105 = 147，所以147和105的最大公约数也是21。在这个过程中，较大的数缩小了，所以继续进行同样的计算可以不断缩小这两个数直至其中一个变成零。这时，所剩下的还没有变成零的数就是两数的最大公约数。  
例如，计算a = 1071和b = 462的最大公约数的过程如下：从1071中不断减去462直到小于462（可以减2次，即商q0 = 2），余数是147：  
1071 = 2 × 462 + 147.  
然后从462中不断减去147直到小于147（可以减3次，即q1 = 3），余数是21：  
462 = 3 × 147 + 21.  
再从147中不断减去21直到小于21（可以减7次，即q2 = 7），没有余数：  
147 = 7 × 21 + 0.  
此时，余数是0，所以1071和462的最大公约数是21。

-----------------------------------------------------------------------------

要求编写两个函数：

1. 求最大公约数的函数gcd()。gcd()的返回值是两个参数的最大公约数，函数原型如下：  
int gcd(int, int);

2. 求最小公倍数的函数lcm()。lcm()的返回值是两个参数的最小公倍数，函数原型如下：  
int lcm(int, int);

函数gcd()和lcm()的调用格式见“Append Code”。

**Input**

输入为多行，每行有一对非负整数a,b，且a\*b不会超出int类型的数据范围。输入至EOF结束。

**Output**

每行输出一对a,b的最大公约数和最小公倍数，顺序与输入对应。  
  
从整除定义出发：若a整除b（b除以a没有余数），则b是a的倍数，a是b的约数，这里要求b不为0。因此0是任意整数的倍数（任意整数都是0的约数），但是0不能是约数。

**Sample Input**

1 1

2 3

2 2

3 2

4 6

7 5

12 6

18 9

24 36

**Sample Output**

1 1

1 6

2 2

1 6

2 12

1 35

6 12

9 18

12 72

**HINT**

 参看系统首页上的“Append Code”使用说明，讨论版（Web Board）上也有。

**Problem C: 编写函数：素数的判定 (Append Code)**

Time Limit: 3 Sec  Memory Limit: 16 MB  
Submit: 22560  Solved: 7988  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=2&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1233)]

**Description**

一个正整数n是素数：除了1和n不能被其他数整除。1不是素数。

-----------------------------------------------------------------------------

编写一个函数用于判定一个正整数是不是素数。

用C语言实现：append.c中函数原型为

int is\_primer(int num);

功能：若num为素数返回1，否则返回0。

用C++实现：append.cc中函数原型为

bool isPrimer(int num);

功能：若num为素数返回true，否则返回false。

函数的调用格式见“Append Code”。

**Input**

每行输入一个正整数n。

n不会超出int类型数据范围。

输入最多不超过20个正整数。

**Output**

若n为素数，输出“YES”，否则输出“NO”。

**Sample Input**

1

2

3

4

5

6

7

8

9

2147483647

**Sample Output**

NO

YES

YES

NO

YES

NO

YES

NO

NO

YES

**HINT**

如果你是按照从小到大的顺序选择除数去判定一个数，那么你超时的原因多半是用了太多不必要的除数。程序需要优化，减少循环的次数。

**Problem D: 编写函数：求累加和 (Append Code)**

Time Limit: 1 Sec  Memory Limit: 2 MB  
Submit: 9276  Solved: 6186  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=3&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1375)]

**Description**

给出两整数min和max，求min到max之间的数的累加和。

-----------------------------------------------------------------------------

编写一个函数add()。函数原型为

int add(int n);

功能：每次调用都把参数n的值累加起来，并返回。

函数add()的调用格式见“Append Code”。

**Input**

输入为两个整数min和max，且max>=min。

**Output**

min和max之间所有整数的累加和，包括min和max。

**Sample Input**

1 10

**Sample Output**

55

**HINT**

 add()函数里要用静态变量来存储累加和。当然用全局变量也行，不过还是练练怎么用静态变量吧。

**Problem E: 判断三角形的性质**

Time Limit: 1 Sec  Memory Limit: 2 MB  
Submit: 12720  Solved: 5801  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=4&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1097)]

**Description**

给出三角形的三边长度，判断是什么三角形。

**Input**

输入为多行，每行3个很小的非负整数，表示一个三角形的三条边的长度。当输入的三角形边长为0时表示输入结束。

**Output**

输出为多行，每行对应一个输入的三条边。

若输入的三条边构成等边三角形，输出“a equileteral triangle”；

若输入的三条边构成等腰三角形，输出“a isosceles triangle”；

若输入的三条边构成直角三角形，输出“a right triangle”；

若输入的三条边构成三角形，但不是以上三种，则输出“a triangle”；

若输入的三条边不构成，输出“not a triangle”。

**Sample Input**

1 1 1

1 1 2

2 2 3

2 3 4

3 4 5

0 0 0

**Sample Output**

a equileteral triangle

not a triangle

a isosceles triangle

a triangle

a right triangle

**HINT**

判断是否直角三角形可用勾股定理。另外，分析样例中得出正确的判断顺序。

**Problem F: Search Problem**

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 13354  Solved: 6236  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=5&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1458)]

**Description**

查找（Search）也翻译成搜索，是计算机科学中的一个很大的子类，也是一种计算机的最常见应用。其核心是利用现代电子计算机的存储能力将数据收集起来，然后用某种查找（搜索）算法在有效的时间内，把需要的数据找出来。

查找（搜索）算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或所有的可能情况，从而求出问题的解的一种方法。查找（搜索）算法按照操作次序可以分成三大类：顺序查找（深度优先、广度优先）、二分查找、直接计算存储位置（如Hash表）。针对应用的性能要求和问题规模，选择不同的查找（搜索）算法。

在解决查找（搜索）的问题时，首先考虑数据组织与结构的问题，即在程序中用何种形式存储数据的。一般根据数据集中数据元素间的相互关系：一对一、一对多、多对多，分别组织成：有序表（也叫线性表）、树、图。这就是构造数据抽象的过程。

利用数据元素间存储位置的顺序关系来组织的结构一般称作数组（Array）、也做列表（list）或向量（Vector），所有的高级程序设计语言都会支持这种存储结构。计算机的内存储器就是按照这种方式来组织的，因此这也是计算机中最基本的存储结构。

数组是这样一种描述一对一关系的数据组织结构：任何一个数据元素都有一个固定的存储位置，每个位置前、后各仅有一个相邻的存储位置。数组中元素的存储位置之间满足一对一的线性序关系，因此可以用自然数来描述每个存储位置上的数据元素，这些自然数就是数组元素的唯一标识——下标。

数组的存储组织结构决定了一个事实：易于存取数据元素、难于改变存储结构。存取数组中数据元素的操作代价可以不计，修改数组中数据元素的代价非常大。因此，数组适用于一次性把数据存储好，不改变或者很少改变数据元素间相互关系的应用中，不适用于那些需要边使用数据边修改结构的问题。

利用数组解决数据查找（搜索）问题的基本步骤是：把数据存储到数组中，按合理的操作次序把想要的数据找出来。关键点是：合理利用下标和数据元素之间的对应关系，解决应该存储什么、以什么顺序存储的问题。

----------------------------------------------------------------------------------------------------------------

你要解决的问题是：在给定的一个整数序列中，按照要求解决一个查找问题。

**Input**

输入分为两部分：

第一部分是第一行，有N+1个整数，第一个正整数为N（N<=10000），表示后跟一个有N个元素的整数序列。

第二部分从第二行开始直到EOF结束，每行为一个整数i。第二部分最多不超过1000行。

**Output**

输出为多行，与输入的第2部分顺序对应，每行输出为：

从输入的整数序列中找到第i个整数并输出它的值，若i不在整数序列的下标范围（1～N）内则输出“OUT OF RANGE”。

**Sample Input**

10 1 2 3 2 3 4 3 3 5 0

1

2

3

5

10

100

**Sample Output**

1

2

3

3

0

OUT OF RANGE

**HINT**

**Problem G: Search Problem (II)**

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 9624  Solved: 5516  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=6&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1459)]

**Description**

查找（Search）也翻译成搜索，是计算机科学中的一个很大的子类，也是一种计算机的最常见应用。其核心是利用现代电子计算机的存储能力将数据收集起来，然后用某种查找（搜索）算法在有效的时间内，把需要的数据找出来。

查找（搜索）算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或所有的可能情况，从而求出问题的解的一种方法。查找（搜索）算法按照操作次序可以分成三大类：顺序查找（深度优先、广度优先）、二分查找、直接计算存储位置（如Hash表）。针对应用的性能要求和问题规模，选择不同的查找（搜索）算法。

在解决查找（搜索）的问题时，首先考虑数据组织与结构的问题，即在程序中用何种形式存储数据的。一般根据数据集中数据元素间的相互关系：一对一、一对多、多对多，分别组织成：有序表（也叫线性表）、树、图。这就是构造数据抽象的过程。

利用数据元素间存储位置的顺序关系来组织的结构一般称作数组（Array）、也做列表（list）或向量（Vector），所有的高级程序设计语言都会支持这种存储结构。计算机的内存储器就是按照这种方式来组织的，因此这也是计算机中最基本的存储结构。

数组是这样一种描述一对一关系的数据组织结构：任何一个数据元素都有一个固定的存储位置，每个位置前、后各仅有一个相邻的存储位置。数组中元素的存储位置之间满足一对一的线性序关系，因此可以用自然数来描述每个存储位置上的数据元素，这些自然数就是数组元素的唯一标识——下标。

数组的存储组织结构决定了一个事实：易于存取数据元素、难于改变存储结构。存取数组中数据元素的操作代价可以不计，修改数组中数据元素的代价非常大。因此，数组适用于一次性把数据存储好，不改变或者很少改变数据元素间相互关系的应用中，不适用于那些需要边使用数据边修改结构的问题。

利用数组解决数据查找（搜索）问题的基本步骤是：把数据存储到数组中，按合理的操作次序把想要的数据找出来。关键点是：合理利用下标和数据元素之间的对应关系，解决应该存储什么、以什么顺序存储的问题。

----------------------------------------------------------------------------------------------------------------

你要解决的问题是：在给定的一个整数序列中，按照要求解决一个查找（搜索）问题。

**Input**

输入分为两部分：第一部分是第一行，有N+1个整数，第一个正整数为N（N<=10000），表示后跟一个有N个元素的整数序列。

第二部分从第二行开始直到EOF结束，每行为一个整数k。

**Output**

输出为多行，与输入的第2部分顺序对应，每行输出为：找到整数序列中第一个值为k的整数并输出它的下标（范围是1～N），若找不到则输出“NOT FOUND”。

**Sample Input**

10 1 2 3 2 3 4 3 3 5 0

1

2

3

5

10

100

**Sample Output**

1

2

3

9

NOT FOUND

NOT FOUND

**HINT**

**Problem H: Search Problem (III)**

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 6490  Solved: 4726  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=7&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1460)]

**Description**

查找（Search）也翻译成搜索，是计算机科学中的一个很大的子类，也是一种计算机的最常见应用。其核心是利用现代电子计算机的存储能力将数据收集起来，然后用某种查找（搜索）算法在有效的时间内，把需要的数据找出来。

查找（搜索）算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或所有的可能情况，从而求出问题的解的一种方法。查找（搜索）算法按照操作次序可以分成三大类：顺序查找（深度优先、广度优先）、二分查找、直接计算存储位置（如Hash表）。针对应用的性能要求和问题规模，选择不同的查找（搜索）算法。

在解决查找（搜索）的问题时，首先考虑数据组织与结构的问题，即在程序中用何种形式存储数据的。一般根据数据集中数据元素间的相互关系：一对一、一对多、多对多，分别组织成：有序表（也叫线性表）、树、图。这就是构造数据抽象的过程。

利用数据元素间存储位置的顺序关系来组织的结构一般称作数组（Array）、也做列表（list）或向量（Vector），所有的高级程序设计语言都会支持这种存储结构。计算机的内存储器就是按照这种方式来组织的，因此这也是计算机中最基本的存储结构。

数组是这样一种描述一对一关系的数据组织结构：任何一个数据元素都有一个固定的存储位置，每个位置前、后各仅有一个相邻的存储位置。数组中元素的存储位置之间满足一对一的线性序关系，因此可以用自然数来描述每个存储位置上的数据元素，这些自然数就是数组元素的唯一标识——下标。

数组的存储组织结构决定了一个事实：易于存取数据元素、难于改变存储结构。存取数组中数据元素的操作代价可以不计，修改数组中数据元素的代价非常大。因此，数组适用于一次性把数据存储好，不改变或者很少改变数据元素间相互关系的应用中，不适用于那些需要边使用数据边修改结构的问题。

利用数组解决数据查找（搜索）问题的基本步骤是：把数据存储到数组中，按合理的操作次序把想要的数据找出来。关键点是：合理利用下标和数据元素之间的对应关系，解决应该存储什么、以什么顺序存储的问题。

----------------------------------------------------------------------------------------------------------------

你要解决的问题是：在给定的一个整数序列中，按照要求解决一个查找（搜索）问题。

**Input**

输入分为两部分：第一部分是第一行，有N+1个整数，第一个正整数为N（N<=10000），表示后跟一个有N个元素的整数序列。

第二部分从第二行开始直到EOF结束，每行为一个整数k。

**Output**

输出为多行，与输入的第2部分顺序对应，每行输出为：找到整数序列中最后一个值为k的整数并输出它的下标（范围是1～N），若找不到则输出“NOT FOUND”。

**Sample Input**

10 1 2 3 2 3 4 3 3 5 0

1

2

3

5

10

100

**Sample Output**

1

4

8

9

NOT FOUND

NOT FOUND

**HINT**

**Problem I: Search Problem (IV)**

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 8389  Solved: 4901  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=8&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1461)]

**Description**

查找（Search）也翻译成搜索，是计算机科学中的一个很大的子类，也是一种计算机的最常见应用。其核心是利用现代电子计算机的存储能力将数据收集起来，然后用某种查找（搜索）算法在有效的时间内，把需要的数据找出来。

查找（搜索）算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或所有的可能情况，从而求出问题的解的一种方法。查找（搜索）算法按照操作次序可以分成三大类：顺序查找（深度优先、广度优先）、二分查找、直接计算存储位置（如Hash表）。针对应用的性能要求和问题规模，选择不同的查找（搜索）算法。

在解决查找（搜索）的问题时，首先考虑数据组织与结构的问题，即在程序中用何种形式存储数据的。一般根据数据集中数据元素间的相互关系：一对一、一对多、多对多，分别组织成：有序表（也叫线性表）、树、图。这就是构造数据抽象的过程。

利用数据元素间存储位置的顺序关系来组织的结构一般称作数组（Array）、也做列表（list）或向量（Vector），所有的高级程序设计语言都会支持这种存储结构。计算机的内存储器就是按照这种方式来组织的，因此这也是计算机中最基本的存储结构。

数组是这样一种描述一对一关系的数据组织结构：任何一个数据元素都有一个固定的存储位置，每个位置前、后各仅有一个相邻的存储位置。数组中元素的存储位置之间满足一对一的线性序关系，因此可以用自然数来描述每个存储位置上的数据元素，这些自然数就是数组元素的唯一标识——下标。

数组的存储组织结构决定了一个事实：易于存取数据元素、难于改变存储结构。存取数组中数据元素的操作代价可以不计，修改数组中数据元素的代价非常大。因此，数组适用于一次性把数据存储好，不改变或者很少改变数据元素间相互关系的应用中，不适用于那些需要边使用数据边修改结构的问题。

利用数组解决数据查找（搜索）问题的基本步骤是：把数据存储到数组中，按合理的操作次序把想要的数据找出来。关键点是：合理利用下标和数据元素之间的对应关系，解决应该存储什么、以什么顺序存储的问题。

----------------------------------------------------------------------------------------------------------------

你要解决的问题是：在给定的一个整数序列中，按照要求解决一个查找（搜索）问题。

**Input**

输入分为两部分：第一部分是第一行，有N+1个整数，第一个正整数为N（N<=10000），表示后跟一个有N个元素的整数序列。

第二部分从第二行开始直到EOF结束，每行为一个整数i。

**Output**

输出为多行，与输入的第2部分顺序对应，每行输出为：

若i>0，从输入的整数序列中找到第i个整数并输出它的值，若i<0，从输入的整数序列中找到倒数第i个整数并输出它的值，若i不在整数序列的下标范围（1～N和-N～-1）内则输出“OUT OF RANGE”。

**Sample Input**

10 1 2 3 2 3 4 3 3 5 0

1

-2

3

-5

10

100

**Sample Output**

1

5

3

4

0

OUT OF RANGE

**HINT**

**Problem J: Search Problem (V)**

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 10048  Solved: 4747  
[[Submit](http://192.168.119.211/JudgeOnline/submitpage.php?cid=4637&pid=9&langmask=1022)][[Status](http://192.168.119.211/JudgeOnline/problemstatus.php?id=1462)]

**Description**

查找（Search）也翻译成搜索，是计算机科学中的一个很大的子类，也是一种计算机的最常见应用。其核心是利用现代电子计算机的存储能力将数据收集起来，然后用某种查找（搜索）算法在有效的时间内，把需要的数据找出来。

查找（搜索）算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或所有的可能情况，从而求出问题的解的一种方法。查找（搜索）算法按照操作次序可以分成三大类：顺序查找（深度优先、广度优先）、二分查找、直接计算存储位置（如Hash表）。针对应用的性能要求和问题规模，选择不同的查找（搜索）算法。

在解决查找（搜索）的问题时，首先考虑数据组织与结构的问题，即在程序中用何种形式存储数据的。一般根据数据集中数据元素间的相互关系：一对一、一对多、多对多，分别组织成：有序表（也叫线性表）、树、图。这就是构造数据抽象的过程。

利用数据元素间存储位置的顺序关系来组织的结构一般称作数组（Array）、也做列表（list）或向量（Vector），所有的高级程序设计语言都会支持这种存储结构。计算机的内存储器就是按照这种方式来组织的，因此这也是计算机中最基本的存储结构。

数组是这样一种描述一对一关系的数据组织结构：任何一个数据元素都有一个固定的存储位置，每个位置前、后各仅有一个相邻的存储位置。数组中元素的存储位置之间满足一对一的线性序关系，因此可以用自然数来描述每个存储位置上的数据元素，这些自然数就是数组元素的唯一标识——下标。

数组的存储组织结构决定了一个事实：易于存取数据元素、难于改变存储结构。存取数组中数据元素的操作代价可以不计，修改数组中数据元素的代价非常大。因此，数组适用于一次性把数据存储好，不改变或者很少改变数据元素间相互关系的应用中，不适用于那些需要边使用数据边修改结构的问题。

利用数组解决数据查找（搜索）问题的基本步骤是：把数据存储到数组中，按合理的操作次序把想要的数据找出来。关键点是：合理利用下标和数据元素之间的对应关系，解决应该存储什么、以什么顺序存储的问题。

----------------------------------------------------------------------------------------------------------------

你要解决的问题是：在给定的一个整数序列中，按照要求解决一个查找（搜索）问题。

**Input**

输入分为两部分：第一部分是第一行，有N+1个整数，第一个正整数为N（N<=10000），表示后跟一个有N个元素的整数序列。

第二部分从第二行开始直到EOF结束，每行为一个整数K。

**Output**

输出为多行，与输入的第2部分顺序对应，每行输出为：找到整数序列中所有值为k的整数并按从小到大输出它们的下标（范围是1～N），若找不到则输出“NOT FOUND”。

**Sample Input**

10 1 2 3 2 3 4 3 3 5 0

1

2

3

5

10

100

**Sample Output**

1

2 4

3 5 7 8

9

NOT FOUND

NOT FOUND

**HINT**