hihoCoder

hiho一下 第九十二周 已经报名

正在进行： 6天22小时19分钟19秒

题目1 : 数论一・Miller-Rabin质数测试

时间限制:10000ms
单点时限:1000ms
内存限制:256MB

描述

小Hi和小Ho最近突然对密码学产生了兴趣，其中有个叫RSA的公钥密码算法。RSA算法的计算过程中，需要找一些很大的质数。

小Ho：要如何来找出足够大的质数呢？

小Hi：我倒是有一个想法，我们可以先随机一个特别大的初始奇数，然后检查它是不是质数，如果不是就找比它大2的数，一直重复，直到找到一个质数为止。

小Ho：这样好像可行，那我就这么办吧。

过了一会儿，小Ho拿来了一张写满数字的纸条。

小Ho：我用程序随机生成了一些初始数字，但是要求解它们是不是质数太花时间了。

小Hi：你是怎么做的啊？

说着小Hi接过了小Ho的纸条。

小Ho：比如说我要检测数字n是不是质数吧，我就从2开始枚举，一直到sqrt(n)，看能否被n整除。

小Hi：那就对了。你看纸条上很多数字都是在15、16位左右，就算开方之后，也有7、8位的数字。对于这样大一个数字的循环，显然会很花费时间。

小Ho：那有什么更快速的方法么？

小Hi：当然有了，有一种叫做Miller-Rabin质数测试的算法，可以很快的判定一个大数是否是质数。

×

提示：Miller-Rabin质数测试

小Hi：这种质数算法是基于费马小定理的一个扩展，首先我们要知道什么是费马小定理：

```
费马小定理：对于质数p和任意整数a，有a^p ≡ a(mod p)(同余)。反之，若满足a^p ≡ a(mod p)，p也有很大概率为质数。
将两边同时约去一个a，则有a^(p-1) ≡ 1(mod p)
```

也即是说：假设我们要测试n是否为质数。我们可以随机选取一个数a，然后计算a^(n-1) mod n，如果结果不为1，我们可以100%断定n不是质数。

否则我们再随机选取一个新的数a进行测试。如此反复多次，如果每次结果都是1，我们就假定n是质数。

该测试被称为Fermat测试。需要注意的是：Fermat测试不一定是准确的，有可能出现把合数误判为质数的情况。

Miller和Rabin在Fermat测试上，建立了Miller-Rabin质数测试算法。

与Fermat测试相比，增加了一个二次探测定理：

```
如果p是奇素数，则 x^2 ≡ 1(mod p)的解为 x ≡ 1 或 x ≡ p - 1(mod p)
```

如果a^(n-1) ≡ 1 (mod n)成立，Miller-Rabin算法不是立即找另一个a进行测试，而是看n-1是不是偶数。如果n-1是偶数，另u=(n-1)/2，并检查是否满足二次探测定理即a^u ≡ 1 或 a^u ≡ n - 1(mod n)。

举个Matrix67 Blog上的例子，假设n=341，我们选取的a=2。则第一次测试时，2^340 mod 341=1。由于340是偶数，因此我们检查2^170，得到2^170 mod 341=1，满足二次探测定理。同时由于170还是偶数，因此我们进一步检查2^85 mod 341=32。此时不满足二次探测定理，因此可以判定341不为质数。

将这两条定理合起来，也就是最常见的Miller-Rabin测试。

但一次MR测试仍然有一定的错误率。为了使我们的结果尽可能的正确，我们需要进行多次MR测试，这样可以把错误率降低。

写成伪代码为：

```
Miller-Rabin(n):
	If (n <= 2) Then
		If (n == 2) Then
			Return True
		End If
		Return False
	End If

	If (n mod 2 == 0) Then
		// n为非2的偶数，直接返回合数
		Return False
	End If
```

```
    // 我们先找到的最小的a^u，再逐步扩大到a^(n-1)

    u = n - 1; // u表示指数
    while (u % 2 == 0)
        u = u / 2
    End While // 提取因子2

    For i = 1 .. S // S为设定的测试次数
        a = rand_Number(2, n - 1) // 随机获取一个2~n-1的数a
        x = a^u % n
        While (u < n)
            // 依次次检查每一个相邻的 a^u, a^2u, a^4u, ... a^(2^k*u)是否满足
二次探测定理
            y = x^2 % n
            If (y == 1 and x != 1 and x != n - 1)   // 二次探测定理
                // 若y = x^2 ≡ 1(mod n)
                // 但是 x != 1 且 x != n-1
                Return False
            End If
            x = y
            u = u * 2
        End While
        If (x != 1) Then        // Fermat测试
            Return False
        End If
    End For
    Return True
```

值得一提的是，Miller-Rabin每次测试失误的概率是1/4；进行S次后，失误的概率是4^(-S)。

小Hi：那么小Ho，你能计算出这个算法的时间复杂度么？

小Ho：恩，每一次单独的MR测试，需要O(log n)的时间。一共要进行S次MR测试，也就是O(Slog n)。

小Hi：没错，这样就能够在很短的时间内完成质数的测试了。当然如果你还是不放心，可以把S的值设定的更高一点。

小Ho：好！这样就能够顺利的找到大质数了。

本题的提示参考了Matrix67的Blog和wikipedia的词条。

Matrix67的Blog有更多的细节描写。Wiki中的伪代码比上文中的简洁一些，并且有介绍了一些小技巧：比如如果n<2^64，只用选取a=2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37做测试即可

Close

输入

第1行：1个正整数t，表示数字的个数，10≤t≤50

第2..t+1行：每行1个正整数，第i+1行表示正整数a[i]，2≤a[i]≤10^18

输出

第1..t行：每行1个字符串，若a[i]为质数，第i行输出"Yes"，否则输出"No"

样例输入

```
3
3
7
9
```

样例输出

```
Yes
Yes
No
```