信息学奥赛笔记19

质数的判断，埃氏筛法求质数。

质数(Prime Number)

在数学中，质数是个想当重要的概念，它也被称为素数，指的是因数只包含`1`和它本身的数。

`30`以内的质数有：

`2 3 5 7 11 13 17 19 23 29`。

在信息学奥赛中，我们需要会两个有关质数的操作，**判断一个数是否是质数**and**通过筛法求出n以内的质数。**

判断质数

首先，我们能够想到最简单的判断质数的方法是什么？就是按照质数的定义，看它的因数是不是只有`1`和它本身，那我们可以跳过`1`和它本身然后一个个去取模运算，如果发现结果为`0`，说明正好可以被这个数整除，所以这个数就不是质数`return 0`，反之则可以在最后说明它是一个质数`return 1`。

```cpp
#include <iostream>
using namespace std;
long long x;
bool isPrime(long long x) {
    for (int i = 2; i < x; i++) {
        if (x % i == 0) return 0;
    }
    return 1;
}
int main() {
    cin >> x;
    cout << (isPrime(x)? "Yes" : "No") << endl;
    return 0;
}
```

判断质数根号优化

那么我们来思考一下`144`这个数的因数，有：

`1 2 3 4 6 8 9 12 16 18 24 36 48 72 144`

那么我们可以发现，一个数的因数都是成对出现的。

因为`1 × 144 = 144`，所以`1`和`144`都是`144`的因数。

`2 × 72 = 144`，所以`2`和`72`都是`144`的因数。

那么最终，一边增大，一边减少，正好相等的点在哪里？

是不是`12 × 12 = 144`此时两边进行到中间了，在数学上我们定义如果`a × a = s`则称为$\sqrt{s}=a$，根号`s`等于`a`，`a`是`s`的根号。

所以对于任何一个数来说，我们在寻找它的因数的时候，只需要找到这个数的根号就可以完成对这个数的检验了。对于一个10^{16}的数来说，它的根号大约为10^8，也就是说，在$1s$内可以完成计算，如果我们像之前那样循环10^{16}的话，会导致超时。

在C++中，我们可以使用求根函数$sqrt$，它在`<cmath>`库中

```
#include <iostream>
#include <cmath>
using namespace std;
long long x;
bool isPrime(long long x) {
    long long n = sqrt(x);
    for (int i = 2; i <= n; i++) {
        if (x % i == 0) return 0;
    }
    return 1;
}
int main() {
    cin >> x;
    cout << (isPrime(x)? "Yes" : "No") << endl;
    return 0;
}
```

六倍定理优化

对于`long long`范围内极端大的一些数来说，即使遍历到它的根号，也就是10^{18}数的根号为10^9还是会导致超时，所以我们的根号判断仍然没办法满足所有的情况。在此基础之上，我们又需要多一步优化操作。

在这里老师可以告诉各位同学们结论。所有的质数都分布在六的倍数附近，换言之一个数`x`如果它是质数的话$x\%6==1$或者$x\%6==5$。

而我们在取模的时候，我们可以思考一下。如果$x\%2!=0$那么还有没有必要算$x\%4$，或者$x\%6$。因为`2`是`4`和`6`的因数，所以既然`x`不是`2`的倍数，那它一定也不是`4`和`6`的倍数，所以即使我们遍历了$\sqrt{x}$遍，但仍然有非常多的数是被浪费的。

那么同理，既然质数分布在`6`的倍数附近，那我们也可以对`6`的倍数附近的数来取模，这样我们就可以避免每次都模重复因数的数。

代码的写法如下：

```
#include <iostream>
#include <cmath>
using namespace std;
long long x;
bool isPrime(long long x) {
    if (x == 1) return 0;  //特判1
    if (x == 2 || x == 3) return 1;//特判2 3
    if (x % 6 != 1 && x % 6 != 5) return 0; //如果对6取模不是1和5，直接return
0;
    long long n = sqrt(x);
    for (int i = 5; i <= n; i += 6) { //我们从1 × 6 - 1开始
```

```
        if (x % i == 0 || x % (i + 2) == 0) return 0;
        //每次模k × 6 - 1 和k × 6 + 1，这样就可以包含从1 - 根号x的所有情况了。
    }
    return 1;
}
int main() {
    cin >> x;
    cout << (isPrime(x)? "Yes" : "No") << endl;
    return 0;
}
```

这样做最坏的情况下，我们需要进行的循环次数为$\frac{\sqrt{10^{18}}}{6} \approx 1.6 \times 10^8$。程序就可以在近似$1s$的情况下计算出来了。六倍定理优化后的判断质数的方法也是唯一一个可以涵盖整个`long long`范围的判断一个数是否是质数的方法了。

埃氏筛

有的时候，我们不仅仅需要判断一个数是否是质数，而是需要将某个范围内所有的质数全部求出来，那么每个数都判断一次，还是太慢了，所以我们需要一个可以一次性一网打尽很多数的方法。

埃氏筛最基本的操作是，每次遇到一个数，先判断它是否被筛除，如果没有则加入进质数库中。

然后将当前这个数一直自增，直到触碰到范围为止，都标记为合数，被筛除。

[U414602] 埃拉托斯特尼筛法(模板)

题目描述

输入一个正整数n，输出$1 - n$内所有的质数。

输入格式

一个正整数n。

输出格式

若干行，表示质数表。

样例 #1

样例输入 #1

```
6
```

样例输出 #1

```
2 3 5
```

提示

对于100%的数据，有$1 \leq n \leq 10^7$。

```cpp
#include<iostream>
using namespace std;
int a[10000000];
long long n;
int main() {
    cin >> n;
    for (int i = 2; i <= n; i++) {
        if (a[i]) continue;
        cout << i <<" ";
        long long j = i;
        while (i * j <= n) {
            a[i * j] = true;
            j++;
        }
    }
    return 0;
}
```

[U414528]判断质数

题目描述

输入两个正整数，请输出它是否是质数。

输入格式

两个整数，用空格隔开。

输出格式

`Yes` 或 `No`，每一个答案占一行。

样例 #1

样例输入 #1

```
3 6
```

样例输出 #1

```
Yes
No
```

提示

对于50%的数据，有 $1 \leq n \leq 10^9$。

对于100%的数据，有$1 \leq n \leq 10^{18}$。

```cpp
#include <bits/stdc++.h>
using namespace std;
bool check(long long x) {
    if (x == 1) return 0;
    if (x == 2 || x == 3) return 1;
    if (x % 6 != 1 && x % 6 != 5) return 0;
    long long n = sqrt(x);
    for (int i = 5; i <= n; i += 6) {
        if (x % i == 0 || x % (i + 2) == 0) return 0;
    }
    return 1;
}
int main() {
    long long n, m;
    cin >> n >> m;
    if (check(n)) cout << "Yes" << endl;
    else cout << "No" << endl;
    if (check(m)) cout << "Yes" << endl;
    else cout << "No" << endl;
    return 0;
}
```

月赛题目讲解

[B3943][语言月赛 202403] 雀？雀！

题目描述

可爱的 szm 妹妹迷上了雀魂麻将。在雀魂麻将中，点数的计算规则为：

- 满贯为 5 番，庄家满贯获得 12000 点，其他玩家满贯获得 8000 点。
- 跳满为 6 到 7 番，可以获得满贯点数（即满贯所获得的点数，下同）的 1.5 倍。
- 倍满为 8 到 10 番，可以获得满贯点数的 2 倍。
- 三倍满为 11 到 12 番，可以获得满贯点数的 3 倍。
- 番数为 $13x \sim 13x+12$（x 是正整数）时，称为 x 倍役满，可获得满贯点数的 $4x$ 倍。

按照游戏规则，如果 szm 获得 x 点，第一名就减少 x 点。例如，第一名当前的点数为 35000，szm 当前的点数为 22000，szm 获得 8000 点后，第一名将减少 8000 点，变为 27000，szm 将增加 8000 点，变为 30000。

现在是 All Last（最后一局），szm 妹妹是庄家，她的点数是第二名，你需要找到最小的能使她变为第一名（点数不低于第一名的点数）的番数。

题目描述的雀魂麻将和真实的雀魂麻将有所不同，请以题目描述为准。

输入格式

输入一行两个整数 x, y，分别表示 szm 妹妹的点数和第一名的点数。

输出格式

输出一行一个整数，表示答案。

样例 #1

样例输入 #1

```
10350 18350
```

样例输出 #1

```
5
```

样例 #2

样例输入 #2

```
10050 10060
```

样例输出 #2

```
5
```

提示

数据规模与约定

对于 30% 的数据，$1 \leq y - x \leq 24000$。
对于 60% 的数据，$1 \leq y - x \leq 10^7$。
对于 100% 的数据，$1 \leq x < y \leq 2 \times 10^9$，$1 \leq y - x \leq 10^9$。

思路分析

这道题目做不出来不怪大家，因为这道题的出题人没有说人话。

我们来把这道题目翻译成人话的结果应该是：

番数	得点
5番	12000
6番	18000
8番	24000
11番	36000
13 * n番	48000 * n

那么，由于得点是一个玩家加，一个玩家减，所以得点应该×2才能得到点差，我们可以画出第二张表。

番数	点差
5番	24000
6番	36000
8番	48000
11番	72000
13 * n番	96000 * n

所以，我们可以先计算两个玩家的点差，然后通过这个点差表去比较，来得出需要多少番。

```cpp
#include <iostream>
using namespace std;
int main() {
    long long x, y, diff;
    cin >> x >> y;
    diff = y - x;
    if (diff <= 24000) {
        cout << 5;
    } else if (diff <= 36000) {
        cout << 6;
    } else if (diff <= 48000) {
        cout << 8;
    } else if (diff <= 72000) {
        cout << 11;
    } else {
        long long bonus = diff % 96000, ans = diff / 96000;
        if (bonus) ans++;
        cout << ans * 13;
    }
    return 0;
}
```

[B3944] 传染病

题目背景

新型病毒正在肆虐洛谷。

题目描述

91-DIVOC 正在广泛传播，珂学家 RyanLi 想要探究 91-DIVOC 的传染系数。

第一天有 a 个人被 91-DIVOC 感染，从第二天起，每个感染者都会向 q 个没有感染的人传播 91-DIVOC，使他们变为感染者。

举个例子，如果第一天有 3 人被感染，每个感染者每天向 2 个人传播病毒，那么第二天会有 3×2 个人被感染。第三天会有 $3 \times 2 \times 2$ 个人被感染 $\cdots$ 以此类推。

定义传染系数为每天被感染 91-DIVOC 的人数的乘积，RyanLi 需要你求出 k 天内的传染系数。由于这个数很大，你只需要输出它对 722733748 取模的结果。

输入格式

输入一行三个整数 k, a, q。

输出格式

输出一行一个整数，表示答案。

样例 #1

样例输入 #1

```
3 3 2
```

样例输出 #1

```
216
```

提示

数据规模与约定

对于 20% 的数据，$k \leq 7$，$a = 2$，$q = 2$。
对于 50% 的数据，$k \leq 10^3$。
对于 100% 的数据，$1 \leq k \leq 10^6$，$1 \leq a, q < 722733748$。

思路分析

这道题只需要根据样例列出表格，代码就非常好写了。

	第1天	第2天	第3天
感染人数	3	3 * 2 = 6	6 * 2 = 12
感染系数	3	3 * 6 = 18	18 * 12 = 216

```cpp
#include <iostream>
using namespace std;
int main() {
    long long k, a, q, mod = 722733748;
    cin >> k >> a >> q;
    long long human = a, ans = a;
    for (int i = 1; i < k; i++) {
        human *= q;
        ans *= human;
    }
    cout << ans % mod << endl;
    return 0;
}
```

但是这个代码只能得到`20`分，原因是，我们在过程中，`human`和`ans`变量都被乘出了非常大的值，导致了值溢出，在最后进行取模这个操作会产生问题。

那我们可以通过一个数学定理

$$(a\%k) * (b\%k) = (a * b)\%k$$

所以我们可以把最终$\%mod$改成每次计算的过程中$\%mod$。

```cpp
#include <iostream>
using namespace std;
int main() {
    long long k, a, q, mod = 722733748;
    cin >> k >> a >> q;
    long long human = a, ans = a;
    for (int i = 1; i < k; i++) {
        human = human * q % mod;
        ans = ans * human % mod;
    }
    cout << ans << endl;
    return 0;
}
```