**SỐ NGUYÊN TỐ, ƯỚC, BỘI,…**

# **Lý thuyết**

## Số nguyên tố

Một số tự nhiên *n (n >* 1) là số nguyên tố nếu *n* có đúng hai ước số là 1 và *n*

Ví dụ các số nguyên tố: 2, 3, 5, 7, 11, 13, 17, 19, 23, ...

### Thuật toán kiểm tra tính nguyên tố

**a. Thuật toán 1:** ĐPT(*n*)

Code

bool isPrime(int n) {

if (n < 2) return false;

for (int i = 2; i < n; i++) {

if (n % i == 0) {

return false;

}

}

return true;

}

**b. Thuật toán 2:** ĐPT()

Code

bool isPrime(int n) {

if (n < 2) return false;

for (int i = 2; i <= static\_cast<int>(std::sqrt(n)); ++i) {

if (n % i == 0) {

return false;

}

}

return true;

}

Hoặc

bool isPrime(int n) {

if (n < 2) return false;

for (int i = 2; i \* i <= n; ++i) {

if (n % i == 0) {

return false;

}

}

return true;

}

### Liệt kê các số nguyên tố trong đoạn [1, n]

**Cách 1:** Ta thử kiểm tra tính nguyên tố của i, với i trong đoạn [1, n]

|  |
| --- |
| void ListNto(int n) { |
| for (int i = 2; i <= n; ++i) { |
| if (isPrime(i)) { |
| cout << i << " "; |
| } |
| } |

**Cách 2:** Sử dụng sàng số nguyên tố(cách này chạy nhanh hơn, tuy nhiên để ý bộ nhớ)

|  |
| --- |
| const int MAXN = 1000000; |
| void Sangnto(int N) { |
| bool Prime[MAXN]; |
| std::memset(Prime, false, sizeof(Prime)); |
| for (int i = 2; i <= static\_cast<int>(std::sqrt(N)); ++i) { |
| if (!Prime[i]) { |
| int j = i \* i; |
| while (j <= N) { |
| Prime[j] = true; |
| j += i; |
| } |
| } |
| } |
| for (int i = 2; i <= N; ++i) { |
| if (!Prime[i]) { |
| std::cout << i << " "; |

## Ước số, bội số

### Số lượng ước của một số

Giả sử số *N* được phân tích thành thừa số nguyên tố như sau:



Trong đó:

- *xi*: là số mũ với 

- *pi*: là số nguyên tố với 

Số lượng ước của *N*: *(x1+1) (x2+1)… (xl+1)*

Ví dụ: , số lượng ước của *20* là: ước số (các ước là: 1, 2, 4, 5, 10, 20)

### Tổng các ước số của một số

Giả sử: 

Tổng các ước của N: 

Ví dụ: Tổng các ước của 20: 

### Ước chung lớn nhất và bội chung nhỏ nhất

**a. Ước chung lớn nhất**

+ Thuật toán Euclid:

- Sử dụng thuật toán Euclid để tìm UCLN(a, b): UCLN(a, b) = UCLN(b, a mod b)

|  |
| --- |
| int UCLN(int a, int b) { |
| int du; |
| while (b != 0) { |
| du = a % b; |
| a = b; |
| b = du; |
| } |
| return a; |
| } |

**b. Bội chung nhỏ nhất:** 

# **Bài tập**

## Bài toán 1. Số lượng ước

### Đề bài:

Cho dãy số nguyên dương:a1, a2,…, aM. Đặt N = a1\*a2\*…\*aM. Hãy tính số lượng ước của N.

Ví dụ: N = 2 \* 3 = 6. Số lượng ước của N là 4 tức là 6 có 4 ước: 1, 2, 3, 6.

**Dữ liệu vào:** Từ tệp văn bản DIVISOR.INP gồm:

* Dòng đầu là số M;
* Dòng thứ hai là dãy số nguyên dương M phần tử: a1, a2,…, aM.

*Các số trên một dòng của tệp dữ liệu vào được ghi cách nhau bởi dấu cách.*

**Kết quả:** Ghi ra tệp văn bản DIVISOR.OUT gồm: Một số nguyên duy nhất là kết quả tính được. Kết quả của bài toán chỉ cần in ra phần dư khi chia cho 109+7.

**Ví dụ:**

|  |  |
| --- | --- |
| **DIVISOR.INP** | **DIVISOR.OUT** |
| 2  2 3 | 4 |

**Ràng buộc:** Các số còn lại trong tệp dữ liệu vào có giá trị tuyệt đối không quá 106.

* *Có 60% số test ứng với 60% số điểm của bài có N ≤ 106;*
* *Có 20% số test ứng với 20% số điểm của bài có M ≤ 1000;*
* *Có 20% số test khác ứng với 20% số điểm còn lại của bài.*

### Hướng dẫn

- Gọi 



- Số lượng ước: (x1+1)\* (x2+1)\*… \*(xl+1)

* ***Sub1****: N ≤ 106;*

*d=0;*

*Duyệt* 

*Nếu*  thì inc(d);

*Cout <<d;*

* ***Sub2,3****:*

- Đếm phân phối vào mảng P[106]

Duyệt  php(a[i]);

- Duyệt mảng P để in kết quả

res =1;

*Duyệt* 

Nếu P[i] > 0 thì res=res\*(P[i]+1);

Cout << res;

## Bài toán 2. Trực nhật

### Đề bài

An và Bình là đôi bạn thân học chung lớp. Trong lớp học thì tất cả các bạn đều phải trực nhật và cứ sau một số x ngày nhất định thì bạn đó mới phải trực nhật lại. Biết rằng, ban đầu An và Bình đều trực nhật ngày đầu tiên, và 2 bạn muốn biết sau bao nhiêu ngày thì 2 bạn lại trực nhật chung với nhau và khi đó mỗi bạn đã trực nhật bao nhiêu lần.

***Yêu cầu:*** Em hãy lập trình để trả lời câu hỏi của An và Bình nhé.

***Dữ liệu vào:*** đọc từ file **TN.INP** gồm:

**-** 1 dòng chứa 2 số nguyên x1, x2 (2 ≤ x1, x2 ≤ 109) lần lượt là số ngày mà sau đó An và Bình mới phải trực nhật lại;

***Dữ liệu ra:*** ghi ra file **TN.OUT** gồm:

- Dòng đầu tiên ghi ra số ngày An và Bình cùng nhau trực nhật lại;

- Dòng thứ 2 chứa 2 số lần lượt là số lần một bạn đã trực nhật cho tới lúc 2 bạn cùng trực nhật.

***Ví dụ:***

|  |  |
| --- | --- |
| **TN.INP** | **TN.OUT** |
| 6 4 | 12  2 3 |

### Hướng dẫn

**Với x1, x2 ≤ 109**

- Sử dụng thuật toán Euclid để tìm UCLN(a, b):

UCLN(a, b) = UCLN(b, a mod b)

- Thuật toán:

|  |
| --- |
| int UCLN(int a, int b) { |
| int du; |
| while (b != 0) { |
| du = a % b; |
| a = b; |
| b = du; |
| } |
| return a; |
| } |

- Công thức tính bội số chung nhỏ nhất của a, b:



## Bài toán 3. MCD

### Đề bài:

Ước số chung đặc biệt của hai số nguyên dương a và b là số nguyên dương d sao cho a chia hết cho d, b chia hết cho d, và tổng các chữ số của d là lớn nhất.

Hãy tìm ước số chung đặc biệt của hai số a và b.

**Input: MCD.INP**

* Một dòng duy nhất chứa hai số nguyên a, b (1<a, b < 109).

**Output: MCD.OUT**

* Trong một dòng duy nhất ghi ra tổng các chữ số của ước số chung đặc biệt của hai số a và b.

**Ví dụ:**

|  |  |
| --- | --- |
| **MCD.INP** | **MCD.OUT** |
| 220 440 | 10 |

***Giải thích***

Ước chung của 220 và 440 là 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110, 220. Trong số các ước này số 55 có tổng các chữ số lớn nhất. Do đó ước chung đặc biệt của 220 và 440 là 55. Kết quả in ra là tổng các chữ số của số 55, tức là bằng 10.

* Hạn chế thời gian: 1s
* Hạn chế bộ nhớ: 256MB

### Hướng dẫn:

* Sub1: A, B ≤ 106
* For i=



If *(&&)* res = max(res,)



* Sub2: A, B ≤ 109

Tính U:



* For i=



If () {



x=A/i;

If *()* res = max(res,)



If *()* res = max(res,)



}

## Bài toán 4. K-factor.

### Đề bài:

Cho số nguyên dương *K*, số nguyên dương *N* gọi là *K*-factor nếu *N* có thể viết được bằng tích của các số nguyên dương bé hơn hay bằng *K*.

Cho số *K* và đoạn nguyên dương [*a*, *b*], hãy xác định có bao nhiêu số nguyên dương *K*-factor thuộc đoạn [*a*, *b*].

***Dữ liệu vào*** từ tệp văn bản **KFACTOR.INP** gồm một dòng ghi 3 số nguyên dương *K*, *a*, *b*; mỗi số cách nhau một dấu cách (2 ≤ *K* ≤ 105,1 ≤ *a* ≤ *b* ≤ 2.109, *b* – *a* ≤ 2.106).

***Kết quả*** ghi ra tệp văn bản **KFACTOR.OUT** chỉ gồm một số nguyên dương là số số nguyên dương *K*-factor thuộc đoạn [*a*, *b*].

*Ví dụ:*

|  |  |  |
| --- | --- | --- |
| **KFACTOR.INP** | **KFACTOR.OUT** | **Giải thích** |
| 5 30 40 | 4 | *Có* 4 *số* 5-factor *thuộc đoạn* [30, 40] *là*:  30 = 2\*3\*5  32 = 2\*4\*4  36 = 3\*3\*4  40 = 2\*4\*5 |

*Giới hạn: Có 60% tests với*:2 ≤ *K* ≤ 104,1 ≤ *a* ≤ *b* ≤ 106, *b* – *a* ≤ 104.

### Hướng dẫn

*Subtast* 1: Duyệt.

Duyệt qua tất cả các số *n* trong đoạn [*a*, *b*], với mỗi số *n* ta phân tích *n* ra thừa số nguyên tố, nếu *n* có một thừa số nguyên tố lớn hơn *K* thì *n* không phải là *K*-factor.

res =0;

For (int n=a; i<=b; i++){

x=n;

d=2;

while ((x>1)&&(d<=k)) {

while (x%d==0) x=x/d;

d++;

}

If (x==1) res++;

}

Độ phức tạp: *O*(*K*(*b* – *a*)).

*Subtast* 2: Tính trước các số nguyên tố và kiểm tra tính *K*-factor như cách sàng các số nguyên tố,

* Để phân tích nhanh ta tính trước và lưu vào mãng prime tất cả các số nguyên tố bé hơn hay bằng *K* (dùng sàng Eratosthene).
* *a*, *b* lớn nhưng *b* – *a* nhỏ nên ta có thể lưu các số từ *a* đến *b* vào mãng *v*[0..5000000]
* Với mỗi số nguyên tố *p* ≤ *K*, ta chia tất cả các số chia hết cho *p* trong mãng *v*, để ý rằng *p* số sẽ có một số chia hết cho *p*. Sau khi chia hết tất cả các số *p* ≤ *K* số nào trong *v* bằng 1 (phân tích được bằng các số nguyên tố bé hơn hay bằng *K*) là số *K*-factor.

Sàng nguyên tố mất cỡ *O*(*K*log*K*), có khoảng số nguyên tố bé hơn hay bằng *K* và khi chia, với mỗi số nguyên tố *p* mất lần chia. Tổng cộng độ phức tạp của thuật toán: cỡ *O*(*K*log*K* + ()) với *p*l ≤ *K*.

## Bài toán 5. Bộ số học

### Đề bài:

Cô giáo Hoa đang giảng bài : “Ước số và bội số là một trong những khái niệm quen thuộc trong số học. Với 2 số nguyên *A* và *B* bất kỳ, nếu *A* chia hết cho *B*, ta nói *A* là bội số của *B* và *B* là ước số của *A*. Với một bộ *K* số nguyên dương *a1, a2,…, aK*bất kỳ, ước số chung lớn nhất của chúng là số nguyên *X* lớn nhất thỏa mãn mọi *ai* là bội số của *X*. Tương tự, bội số chung nhỏ nhất của bộ số này là số nguyên *Y* nhỏ nhất thỏa mãn mọi *ai* là ước số của *Y*”. Bất chợt học sinh Phát trong lớp nghĩ ra một bài toán: “Cho 2 dãy số nguyên *p1, p2,…, pM*và *q1, q2,…, qN*. Đặt *P= p1\*p2\*…\*pM* và *Q= q1\*q2\*…\*qN*. Đếm số bộ *K* số nguyên có ước số chung lớn nhất là *P* và bội số chung nhỏ nhất là *Q*”.

**Yêu cầu:** Các bạn là học sinh tuyển chọn của trường hãy giúp Phát giải bài toán này nhé.

**Dữ liệu vào:** Từ tệp văn bản **SOHOC.INP** gồm:

* Dòng đầu tiên gồm ba số nguyên dương *M, N, K;*
* Dòng thứ hai gồm *M* số nguyên dương *p1,p2,…, pM;*
* Dòng thứ ba gồm *N* số nguyên dương *q1, q2,…, qN.*

*Các số trên một dòng của tệp dữ liệu vào được ghi cách nhau bởi dấu cách.*

**Kết quả:** Ghi ra tệp văn bản **SOHOC.OUT** gồm: Một số nguyên duy nhất là số bộ số thỏa mãn. Kết quả của bài toán chỉ cần in ra phần dư khi chia cho *109+9*.

**Ví dụ:**

|  |  |  |
| --- | --- | --- |
| **SOHOC.INP** | **SOHOC.OUT** | ***Giải thích*** |
| 1 2 2  1  2 3 | 4 | P=1; Q=6. 4 bộ 2 số thỏa mãn điều kiện là (1,6), (2,3), (3,2), (6,1). |
| **SOHOC.INP** | **SOHOC.OUT** | ***Giải thích*** |
| 2 1 2  2 5  131 | 0 | P=10; Q=131. Do Q không chia hết cho P nên dễ thấy không tồn tại bộ số nào thỏa mãn. |

**Ràng buộc:** Trong tất cả bộ test: *K≤109*, các số còn lại trong tệp dữ liệu vào có giá trị tuyệt đối không quá *106.*

* *Có 10% số test ứng với 10% số điểm của bài có* *M=N=K=1*
* *Có 30% số test ứng với 30% số điểm của bài có max(P, Q) ≤ 3x103 và K=2;*
* *Có 20% số test ứng với 20% số điểm của bài có max(P, Q) ≤ 106 và K=2;*
* *Có 20% số test ứng với 20% số điểm của bài có max(M, N) ≤ 5x103 và K=2;*
* *Có 20% số test khác ứng với 20% số điểm còn lại của bài.*

### Hướng dẫn:

* **Sub1:** Với *M=N=K=1*

Dùng tính chất chia hết

* **Sub2:** *max(P, Q) ≤ 3x103 và K=2;*

Duyệt 2 for để tìm bộ số thỏa mãn đề bài.

* **Sub3:** *max(P, Q) ≤ 106 và K=2;*

Phân tích ra thừa số nguyên tố.

* **Sub4:** *max(M, N) ≤ 5x103 và K=2;*

Chuẩn bị mảng sàng nguyên tố để giải quyết.

* **Sub5:** Sử dụng kiến thức toán về phương pháp đếm/chỉnh hợp.

dữ liệu ra có nhiều bài toán.

# **Bài tập tự luyện**

## Bài toán 1. FUNCTION

### Đề bài :

Vào thế kỷ trước một nhà toán học đã định nghĩa hàm trên dãy gồm số nguyên dương như sau: với , trong đó là ước chung lớn nhất của các số .

Vài năm sau đó một nhà toán học khác áp dụng hàm trên dãy 1, 1, …, 1 và nhận xét rằng hàm luôn có giá trị bằng 1. Trên cơ sở đó ông ta đưa ra giả thiết là giá trị của hàm luôn là một hằng số mà không phụ thuộc gì vào dãy .

**Yêu cầu:** Với kiến thức toán học và lập trình của mình bạn hãy bác bỏ giả thiết trên bằng cách chỉ ra hàm có thể có nhiều giá trị khác nhau trên dãy cho trước.

**Dữ liệu vào:** Từ tệp văn bản FUNCTION.INP gồm

* Dòng 1 ghi số nguyên dương .
* Dòng 2 ghi số nguyên dương

**Kết quả:** Ghi ra tệp văn bản FUNCTION.OUT gồm một dòng ghi một số là số giá trị khác nhau của hàm trên dãy đã cho.

**Ví dụ:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION.INP** | **FUNCTION.OUT** |  | **FUNCTION.INP** | **FUNCTION.OUT** |
| 4  9 6 2 4 | 6 |  | 4  9 6 3 4 | 5 |

*Ràng buộc:*

* *Có 40% số lượng test thỏa mãn ;*
* *Có 30% số lượng test thỏa mãn ;*
* *Có 30% số lượng test thỏa mãn .*

### Hướng dẫn:

Cho dãy gồm số nguyên dương , bài toán yêu cầu tìm số giá trị khác nhau của hàm: với .

*Thuật toán*: Duyệt mọi dãy con, với mỗi dãy tìm của chúng và lưu lại các giá trị khác nhau. Cuối cùng ghi ra số lượng các giá trị khác nhau đó.

*Nhận xét*: Số lượng các giá trị khác nhau của với tối đa là giá trị.

**Thuật toán** 1: Duyệt mọi dãy, lưu các giá trị khác nhau trong mãng hoặc dùng cấu trúc dữ liệu có thể bỏ các giá trị trùng (set). Độ phức tạp trong trường hợp này là .

**Thuật toán** 2: Để giảm số lần duyệt các dãy con ta có nhận xét sau:

Gọi tập này có nhiều nhất là giá trị. Khi đó việc tìm các giá trị của tập dựa trên tập : . Vậy việc tìm mỗi chỉ mất . Việc lưu kết quả cũng như thuật toán 1. Độ phức tạp .

## Bài toán 2. CSNTO

### Đề bài :

Tìm K chữ số nguyên tố trong đoạn [A, B].

**Yêu cầu:** Cho đoạn [A, B], hãy tìm số lượng số nguyên tố có k chữ số trong đoạn đó.

**Dữ liệu vào:** File văn bản CSNTO.INP

- Gồm một dòng: Chứa K, A và B.

**Dữ liệu ra:** file văn bản CSNTO.OUT

- Gồm một số duy nhất là số lượng số nguyên tố cần tìm.

**Ví dụ:**

|  |  |
| --- | --- |
| **CSNTO.INP** | **CSNTO.OUT** |
| 1 2 10 | 4 |

**Ràng buộc:**

* A, B số nguyên ≤106.

### Hướng dẫn:

Res = 0;//Số lượng số nguyên tố trong đoạn [A, B]

Duyệt i từ a đến b

Nếu isPrime(i) thì

Nếu cs(i) = K thì res ++;

## Bài toán 3. Bảng nguyên tố

### Đề bài:

Cho một bảng các số nguyên dương gồm n dòng m cột (1≤n, m≤500). Một thao tác trên bảng được định nghĩa như sau: chọn một số bất kì trong bảng và tăng giá trị của số đó lên 1. Một số có thể được chọn để thực hiện thao tác trên nhiều hơn 1 lần.

Người ta định nghĩa một bảng là có tính nguyên tố nếu thỏa mãn một trong hai điều kiện sau:

* Trong bảng tồn tại một hàng bất kì mà tất cả các số trong hàng đều là số nguyên tố.
* Trong bảng tồn tại một cột bất kì mà tất cả các số trong cột đều là số nguyên tố.

**Yêu cầu**: Tìm số thao tác biến đổi ít nhất để bảng ban đầu thành bảng có tính nguyên tố.

**Dữ liệu vào** cho từ tệp **MPRIME.INP** gồm:

* Dòng 1 gồm 2 số n, m tương ứng là số dòng và số cột của bảng
* N dòng sau, mỗi dòng gồm m số nguyên dương có giá trị ≤ 106 là các số trong bảng ban đầu. Các số trên một dòng cách nhau một dấu cách.

**Kết quả** đưa ra tệp **MPRIME.OUT** gồm: Một số duy nhất là số thao tác ít nhất cần biến đổi để bảng ban đầu có tính nguyên tố.

**Ví dụ:**

|  |  |
| --- | --- |
| MPRIME.INP | MPRIME.OUT |
| 3 3  1 2 3  4 9 8  2 3 9 | 1 |

**Ràng buộc**:

* *50% số test ứng với 50% số điểm của bài có n≤50*

### Hướng dẫn:

* Cách 1:

+ Duyệt theo hàng, với số x ở hàng i, tìm số nguyên tố gần số x nhất gọi là y. Số thao tác biến đổi số x là t=y-x. Gọi Si là tổng số thao tác biến đổi trên hàng i là . Kết quả tìm được số thao tác biến đổi ít nhất trên hàng là M1=min(Si)

+ Tương tự duyệt theo cột và tìm được số thao tác biến đổi ít nhất trên cột, gọi là M2.

* Kết quả bài toán là MIN(M1,M2)
* Cách 2: Nhận xét các số trong bảng có giá trị lớn nhất 106

+ Sử dụng sàng nguyên tố tính trước các số nguyên tố.

+ Mỗi số trong bảng tìm được khoảng cách số nguyên tố gần nhất số đó.

## Bài toán 4. Ước tự nhiên

### Đề bài:

Một số tự nhiên *N,* nếu tồn tại 2 số tự nhiên *x* và *y* sao cho *N* = *x* \* *y,* thì *x* và *y* là các ước tự nhiên của *N*.

**Yêu cầu:** Cho số tự nhiên *N(N <* 106). Hãy tính số lượng các ước tự nhiên của *N*.

**Dữ liệu vào:** Từ tệp văn bản SLUOC.INP có cấu trúc:

* Dòng đầu tiên ghi số nguyên dương *T* là số bộ dữ liệu;
* T dòng tiếp theo, mỗi dòng chứa 1 số tự nhiên N tương ứng với 1 bộ dữ liệu.

**Kết quả:** Ghi ra tệp văn bản SLUOC.OUT gồm *T* dòng, mỗi dòng ghi một số nguyên *u* là số lượng các ước tự nhiên tương ứng với dữ liệu vào.

**Ví dụ:**

**Ràng buộc:**

| **SLUOC.INP** | **SLUOC.OUT** |
| --- | --- |
| 1 | 4 |
| 10 |  |

| **SLUOC.INP** | **SLUOC.OUT** |
| --- | --- |
| 2 | 2 |
| 3 | 6 |
| 20 |  |

* *Có 30% test ứng với 30% số điểm của bài có T* = 1; *1* ≤ *N* ≤ 103;
* *Có 20% test ứng với 20% số điểm của bài có T* ≤ 10; *1* ≤ *N* ≤ 103;
* *Có 20% test ứng với 20% số điểm của bài có T* ≤ 10; *1* ≤ *N* ≤ 106;
* *Có 30% test khác ứng với 30% số điểm còn lại của bài có T* ≤ 106; *1* ≤ *N* ≤ 106.

### Hướng dẫn:

Số lượng ước tự nhiên của *n*.

**Thuật toán** 1: Với ràng buộc *T ≤10; 1 ≤ n ≤ 103*

Duyệt tất cả các số nguyên i từ 1 tới n, nếu n chia hết i thì tăng số ước. Độ phức tạp trong trường hợp này là *O(T x n)*.

**Thuật toán** 2: Với ràng buộc *T ≤106; 1 ≤ n ≤ 106*

Trong quá trình duyệt các số nguyên *i* từ *1* tới *n*, cần giảm chi phí tính số ước của . Ta có nhận xét với mỗi số =, trong đó là các số nguyên tố thì:

* Công thức tính số ước là:.

Việc xác định các dùng sàng số nguyên tố.

* Việc tính số lượng ước chỉ tốn hằng số c.
* Độ phức tạp trong trường hợp này là *O(T x c).*

## Bài toán 5. Tìm số nguyên gần nhất

### Đề bài:

Tìm số nguyên tố gần nhất nhỏ hơn N (3 ≤ N ≤ 232).

**Dữ liệu vào:** File văn bản **SUPERNTO.INP** gồm

* Dòng đầu tiên chứa số nguyên T là số lượng test. (T ≤10000). T dòng tiếp theo, mỗi dòng chứa một số nguyên N.

**Kết quả:** Ghi ra tệp văn bản **SUPERNTO.OUT** gồm nhiều dòng mỗi dòng tương ứng mỗi test.

**Ví dụ**

|  |  |
| --- | --- |
| **SUPERNTO.INP** | **SUPERNTO.OUT** |
| **3**  **5**  **10**  **17** | **3**  **7**  **13** |

Hạn chế thời gian: 5 giây.

### Hướng dẫn:

**Thuật toán**: Với ràng buộc *T ≤104;* N ≤ 232

N--;

While (not(prime(N))) {N--}

Cout << N<<endl;

* Độ phức tạp trong trường hợp này là *O(T x* *).*