SÀNG NGUYÊN TỐ VÀ SÀNG MỞ RỘNG

1. Số nguyên tố

Số nguyên dương $p > 1$ được gọi là nguyên tố nếu và chỉ nếu p tồn tại đúng 2 ước số là 1 và chính nó. Số nguyên dương lớn hơn 1 và không phải là nguyên tố được gọi là hợp số.

Thuật toán kiểm số nguyên dương n có phải là số nguyên tố.

Thuật toán 1: độ phức tạp $O(n)$

```
bool IsPrime(int n)
{
    int cnt = 0;
    for (int i = 1; i <= n; ++i)
        if (n % i == 0) ++cnt;
    return (cnt == 2);
}
```

Thuật toán 2: độ phức tạp $\mathrm{O}(\sqrt{\mathrm{n}})$

Nhận xét:

- Nếu a là ước dương của n thì luôn tồn tại b cũng là ước dương của n thỏa $a \times b = n (b = n \text{ div } a)$
- Giả sử $1 \leq a \leq b \Longrightarrow a^2 \leq a \times b = n \Longrightarrow 1 \leq a \leq \sqrt{n}$.
- Vì $n > 1$ luôn tồn tại 2 ước dương là 1 và chính nó nên ta chỉ cần kiểm tra nếu tồn một ước của n từ 2 đến $\sqrt{n}$ thì n không phải là số nguyên tố.

Cài đặt

```
bool IsPrime(int n)
{
    if (n < 2) return false;

    int m = (int)sqrt(n);
    for (int i = 2; i <= m; ++i)
        if (n % i == 0) return false;

    return true;
```

```
}
```

Thuật toán 3: tạo sàng nguyên tố

Sàng nguyên tố được ứng dụng trong trường hợp cần kiểm tra tính nguyên tố của nhiều số nguyên. Để tạo một sàng nguyên tố, ta tạo mảng 1 chiều trong đó chỉ số mảng trùng với giá trị của số nguyên tố tại vị trí đó. Với mảng `d` có kiểu `bool` làm sàng nguyên tố, ta có `d[x] = false` nếu x là số nguyên tố, ngược lại `d[x] = true`.

Thuật toán tạo sàng

```
bool d[maxN] = {false};

void Sieve()
{
    d[0] = d[1] = true;
    for (int x = 2; x <= sqrt(maxN); ++x)
        if (d[x] == false)
            for (int y = x; y <= maxN/x; ++y)
               d[y*x] = true;
}
```

Áp dụng: Cho dãy số nguyên $a_1, a_2, \ldots, a_n (|a_i| \leq 10^6)$. Hãy đếm số phần tử của dãy là số nguyên tố.

```
void Solve()
{
    int ans = 0;
    cin>>n;
    for (int i = 1; i <= n; ++i)
    {
        cin>>x;
        if (x > 1 && d[x] == false) ++ans;
    }
    cout<<ans;
}

int main()
{
    Sieve();
    Solve();
    return 0;
}
```

2. Sàng mở rộng và ứng dụng

Kỹ thuật mảng đánh dấu sử dụng trong tạo sàng nguyên tố được ứng dụng để giải một số bài toán cụ thể được trình bày bên dưới đây và được tạm gọi là kỹ thuật sàng mở rộng.

2.1. Ứng dụng 1

Gọi $d(n)$ là số ước số của n, ví dụ $d(6) = 4$. Tính $d(n), \forall n \leq 10^6$.

Nhận xét

- Nếu x là ước dương của n thì luôn tồn tại y cũng là ước dương của n sao cho $x \times y = n$.
- Xét tất cả cặp x, y với $1 \leq x < y$, ta có x, y là 2 ước khác nhau của số nguyên $x \times y$ vì vậy tăng giá trị của $d[x \times y]$ thêm 2.

Cài đặt

```
int d[maxN] = {0};
void Sieve()
{
    for (int x = 1; x <= sqrt(maxN); ++x)
    {
        ++d[x*x]; //x là 1 ước của x*x
        for (int y = x+1; y <= maxN/x; ++y)
            d[x*y] += 2; //x, y là 2 ước khác nhau của x*y (x < y)
    }
}
```

2.2. Ứng dụng 2

Gọi $d(n)$ là tổng tất cả ước của n, ví dụ $d(12) = 1 + 2 + 3 + 4 + 6 + 12 = 28$. Tính $d(n), \forall n \leq 10^6$.

Nhận xét

- Nếu x là ước dương của n thì luôn tồn tại y cũng là ước dương của n sao cho $x \times y = n$.
- Xét tất cả cặp x, y với $1 \leq x < y$, ta có x, y là 2 ước khác nhau của số nguyên $x \times y$ vì vậy tăng giá trị của $d[x \times y]$ thêm giá trị $(x + y)$.
- Do số 1 là ước của bất kỳ số nào nên $d(n)$ được khởi tạo là 1. Hơn nữa ta chỉ xét các ước $< n$ nên x được xét từ 2 trở đi (để tránh xét $y = n/x = n$).

Cài đặt

```
void Sieve()
{
    for (int x = 1; x <= sqrt(maxN); ++x)
    {
        d[x*x] = d[x*x] + x; //x là ước của x^2
        for (int y = x+1; y <= maxN/x; ++y)
            d[x*y] = d[x*y] + x + y; //x, y là 2 ước khác nhau của x*y
    }
}
```

2.3. Ứng dụng 3

Gọi $d(n)$ là tổng tất cả thừa số nguyên tố trong phân tích dạng không lũy thừa của n, ví dụ $d(20) = 2 + 2 + 5 = 9$. Tính $d(n), \forall n \leq 10^6$.

Cài đặt

```
void Sieve()
{
    for (int x = 2; x <= maxN; ++x)
    {
        if (d[x] == 0) //x là số nguyên tố
            for (int y = 1; y <= maxN/x; ++y)
            {
                z = y*x;
                while (z % x == 0)
                {
                    d[x*y] += x; //cộng x vào những vị trí là bội của x
                    z = z/x;
                }
            }
    }
}
```

2.4. Ứng dụng 4

Gọi $d(n)$ là tổng tất cả thừa số nguyên tố khác nhau trong phân tích của n, ví dụ $d(20) = 2 + 5 = 7$. Tính $d(n), \forall n \leq 10^6$.

Nhận xét

- Nếu x là số nguyên tố thì ta cộng thêm x vào tất cả vị trí $z = x \times y$ là bội của x. Do vậy ta có $d(z) > 0$. Ngược lại $d(z) = 0$ thì z là số nguyên tố.
- Vì ta xét các bội của x bắt đầu từ x nên x lặp từ 2 đến `maxN`

Cài đặt

```
void Sieve()
{
    for (int x = 2; x <= maxN; ++x)
    {
        if (d[x] == 0) //x là số nguyên tố
            for (int y = 1; y <= maxN/x; ++y)
                d[y*x] += x; //cộng thêm x vào những vị trí là bội của x
    }
}
```