LỜI GIẢI ĐỀ TIN HỌC TRẺ | VÒNG QUẬN TÂN BÌNH 2025

Tác giả: Hưng (9A2, NGT)

Bài 1:

- **Tóm tắt đề:** cho một hình vuông kích thước $n \times n$ $(1 \leq n \leq 10^{20})$. Hãy đếm xem có bao nhiêu hình vuông con có độ dài cạnh lớn hơn 2 ô vuông. Lấy phần dư của kết quả khi chia cho 2023.

Lời giải:

Một hình vuông con có độ dài cạnh là k có thể được xác định bởi góc dưới bên phải. Nếu tọa độ góc dưới bên phải là (i, j), nghĩa là hàng i, cột j, các cột có thể chứa được hình vuông con đó là $k, k+1, \dots, n$. Tương tự, các hàng có thể chứa hình vuông con đó cũng là $k, k+1, \dots, n$. Có tất cả $n-k+1$ hàng và $n-k+1$ cột. Ta có thể chọn một hàng bất kì ghép với một cột bất kì nên có $(n-k+1) \times (n-k+1)$ cách ghép khác nhau, hay là $(n-k+1)^2$ cách chọn hình vuông con hợp lệ.

Công thức tính số lượng hình vuông con có độ dài cạnh $k \times k$ trong hình vuông kích thước $n \times n$ là:

$$(n-k+1)^2$$

⇨ Công thức tính số lượng hình vuông có độ dài cạnh lớn hơn 2, nghĩa là $k \geq 3$ là:

$$\sum_{k=3}^{n} (n-k+1)^2$$

Thay $k = 3,4,5, \dots$ vào thì kết quả sẽ là: $(n-2)^2 + (n-3)^2 + \cdots + 2^2 + 1^2$

Đây chính là tổng bình phương của $m = n-2$ số nguyên dương đầu tiên. Có công thức là: $\frac{m(m+1)(2m+1)}{6}$

Vậy kết quả là: $\frac{m(m+1)(2m+1)}{6}$ với $m = n-2$.

Có thể dùng Python để trực tiếp tính ra kết quả. **Code:**

```
n=int(input())

m=n-2
print((m*(m+1)*(2*m+1)//6)%2023)
```

(*) Tuy nhiên, với những người sử dụng C++, không có khả năng xử lý số lớn tới 10^{20}, việc xử lý số lớn thủ công rất mất thời gian và dễ sai. Do vậy ta cần sử dụng toán học, nhờ cách số 2 như sau.

Ta định nghĩa nghịch đảo modulo m của b là một số nguyên x thỏa mãn:

$$bx \equiv 1 \ (mod \ m)$$

Số x tồn tại khi b và m là 2 số nguyên tố cùng nhau!

Bây giờ, ta muốn tính phần dư của $\frac{a}{b}$ khi chia cho m. Không có công thức nào để tính trực tiếp nó như phép cộng, trừ và nhân. Nhưng ta để ý thấy:

$$\frac{a}{b} = \frac{a}{b}.1$$

Mà $1 \equiv bx \ (mod \ m)$ với x là nghịch đảo modulo m của b. Thay vào giá trị 1, ta được:

$$\frac{a}{b}.1 \equiv \frac{a}{b}.bx \ (mod \ m)$$

Rút gọn b, ta còn lại:

$$\frac{a}{b}.1 \equiv \frac{a}{b}.bx = ax \ (mod \ m)$$

⇨ $\frac{a}{b} \equiv ax \ (mod \ m)$ với x là nghịch đảo modulo m của b.

Vậy thay vì tính trực tiếp $\frac{a}{b}$ xong chia lấy dư cho m, ta chỉ cần tìm nghịch đảo modulo m của b xong thì kết quả sẽ là ax với x là nghịch đảo modulo m của b. Lúc này ta có thể áp dụng công thức:

$$(u \times v) \ mod \ m = \big((u \ mod \ m) \times (v \ mod \ m)\big) \ mod \ m$$

để tránh tràn số là được.

Quay lại bài chính, để tính kết quả là $\frac{m(m+1)(2m+1)}{6} \ mod \ 2023$. Vì 6 và 2023 nguyên tố cùng nhau nên tồn tại nghịch đảo modulo 2023 của 6. Để tìm số x thỏa $6x \equiv 1 \ (mod \ 2023)$ ta chỉ cần chạy đoạn code sau:

```
#include<bits/stdc++.h>
using namespace std;

signed main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    int b=6,m=2023;
    for(int x=1;;++x){
        if((b*x)%m==1){ // b ≡ 1 (mod m)
            cout<<x<<'\n';
            break;
        }
    }

    return 0;
}
```

Sau khi chạy, kết quả hiển thị ra màn hình là $\boxed{1686}$. Vậy số x ta cần tìm là 1686.

Vậy kết quả giờ sẽ là: $m(m+1)(2m+1) \times 1686$ khi lấy dư cho 2023.

Tuy nhiên do n có thể lên tới 10^{20} nên ta cần lưu nó dưới dạng xâu. Mà trong hệ thập phân, một số $\overline{s_1 s_2 s_3 \dots s_d}$ nào đó có d chữ số có thể được biểu diễn dưới dạng cơ số 10:

$$\overline{s_1 s_2 s_3 \dots s_d} = s_1.10^{d-1} + s_2.10^{d-2} + \cdots + s_{d-1}.10^1 + s_d.10^0$$

Nhận thấy $10^d \ mod\ 2023 = \left(10^{d-1}.10\right) mod\ 2023$ nên ta có thể lấy dư cho m bằng cách lần lượt tính $10^{d-1}\ mod\ 2023$ rồi nhân tiếp với 10 xong lại lấy dư cho 2023 (áp dụng tính chất đồng dư của phép nhân).

Sau đó các chữ số thực chất được công lại với nhau sau khi nhân với một lũy thừa của 10. Do vậy ta có thể áp dụng tiếp tính chất đồng dư của phép cộng để tiếp tục lấy dư cho 2023. Vậy để tính dư số s khi chia cho 2023, ta gọi hàm $get\_num\_mod()$ trong đoạn code sau:

```
int get10mod(int d){
    int res=1; // 10^0=1
    for(int mu=1;mu<=d;++mu){
        res=(res*10)%MOD; // 10^(d-1)*10 mod 2023
    }
    return res;
}

int get_num_mod(string s){
    int res=0;
    int d=s.size();
    for(int mu=d-1;mu>=0;--mu){
        int digit=s[d-mu-1]-'0'; // chuyen doi tu char sang int
        res=(res+digit*get10mod(mu))%MOD;
    }
    return res;
}
```

hoặc có cách nhanh hơn:

```
int get_num_mod(string s){
    int res=0;
    for(int digit:s){
        res=(res*10+digit-'0')%MOD;
    }
    return res;
}
```

Sau đó ta đặt $m = (n-2)\ mod\ 2023$ luôn để tránh tràn số.

Vậy sau tất cả những đau khổ thì ta đã đến phần quan trọng nhất! **Code:**

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;

const int MOD=2023,INV6=1686;

int get_num_mod(string s){
    int res=0;
    for(int digit:s){
        res=(res*10+digit-'0')%MOD;
    }
    return res;
}

signed main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    string num;
    cin>>num;

    int n=get_num_mod(num)-2;
    cout<<((n*(n+1)*(2*n+1)*INV6)%MOD+MOD)%MOD<<'\n';
    return 0;
}
```

- Với bộ test là $n = 123456789012345678$ thì kết quả trả ra là $\boxed{217}$. (!!)

(*) Ngoài ra còn một cách cũng ngắn không kém, đó là áp dụng trực tiếp công thức tính số dư:

$$\frac{a}{b} \mod m = \frac{a \mod (m \times b)}{b}$$

Áp dụng công thức đó vào code ta được ngay kết quả khá dễ dàng. **Code:**

```cpp
#include<bits/stdc++.h>
#define int long long
using namespace std;

const int MOD=2023;
int get_num_mod(string s){
    int res=0;
    for(int digit:s){
        res=(res*10+digit-'0')%MOD;
    }
    return res;
}

signed main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    string num;
    cin>>num;

    int n=(get_num_mod(num)-2+MOD)%MOD;
    cout<<(n*(n+1)*(2*n+1)%(6*MOD)/6)<<'\n';
    return 0;
}
```

Bài 2:

- **Tóm tắt đề:** cho phân số $\frac{a}{b}$ $(a, b \in N^*)$. Hãy tối giản phân số đó.

Lời giải:

Dễ thấy để tối giản phân số thì ta chỉ cần lấy cả tử và mẫu chia cho ƯCLN của chúng. Hãy nói cách khác nếu gọi g là $\gcd(a, b)$ thì phân số tới giản sẽ là: $\frac{a/g}{b/g}$.

Code:

```
#include<bits/stdc++.h>
#define int long long
using namespace std;

signed main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    int a,b;
    cin>>a>>b;

    int g=__gcd(a,b);
    cout<<a/g<<' '<<b/g<<'\n';
    return 0;
}
```

Bài 3:

- **Tóm tắt đề:** cho số A $(0 \leq A \leq 100)$. Tìm 4 chữ số tận cùng của 2^A (nếu 2^A có không nhiều hơn 4 chữ số thì in ra trực tiếp 2^A).

Lời giải:

Để tìm k chữ số tận cùng của một số N nào đó, ta thực chất chỉ cần lấy phần dư của số N đó khi chia cho 10^k. Do vậy ta đưa về bài toán tính 2^A chia lấy dư cho 10000.

Nhắc lại tính chất đồng dư của phép nhân:

$$(u \times v)\ mod\ m = \big((u\ mod\ m) \times (v\ mod\ m)\big)\ mod\ m$$

Thực chất 2^A cũng chỉ là $2 \times 2 \times 2 \times \ldots \times 2$ lặp lại A lần nên ta áp dụng tính chất trên để vừa nhân, vừa lấy dư cho 10000 là được. **Code:**

```
#include<bits/stdc++.h>
#define int long long
using namespace std;

signed main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    int A;
    cin>>A;

    int res=1;
    while(A--){
        res=(res*2)%10000;
    }

    cout<<res<<'\n';
    return 0;
}
```