Number Theory is

a branch of mathematics that deals with the properties and relationships of numbers, especially positive integers.

Contents

- What is Number Theory?
- Even and Odd
- Prime Numbers and Divisors
- Prime Factorization
- GCD and LCM
- Sieve of Eratosthenes

Even and Odd Numbers

Even

An integer n is even, if and only if, n is twice some integer.

e.g. n=2*k

Odd

An integer n is odd, if and only if, n is twice some integer plus 1

e.g. n = 2k+1

Prime and Divisor Numbers

Prime

An integer $n > 1$ is prime, if and only if, for all positive integers r and s, if $n=r \times s$, then $r=1$ or $s=1$

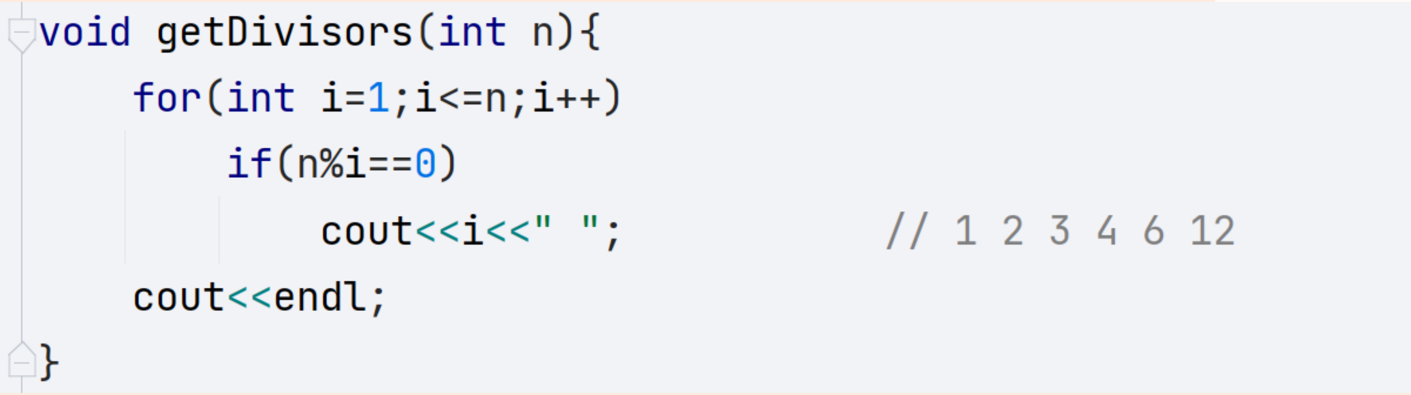
e.g. n=1*r

Divisor

An integer n is divisible by an integer d ($d \neq 0$), if and only if, n equals d times of some integer k

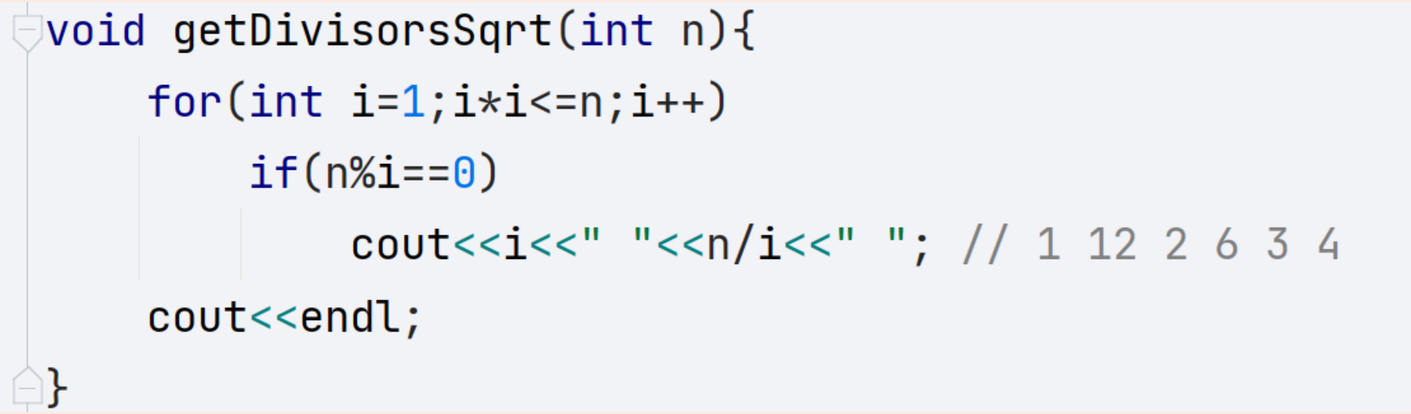
e.g. n = d*k

getDivisors → O(n)



```
void getDivisors(int n){
    for(int i=1;i<=n;i++)
        if(n%i==0)
            cout<<i<<" ";              // 1 2 3 4 6 12
    cout<<endl;
}
```

getDivisorsSqrt → O($\sqrt{n}$)



```
void getDivisorsSqrt(int n){
    for(int i=1;i*i<=n;i++)
        if(n%i==0)
            cout<<i<<" "<<n/i<<" "; // 1 12 2 6 3 4
    cout<<endl;
}
```

Did you notice something wrong?

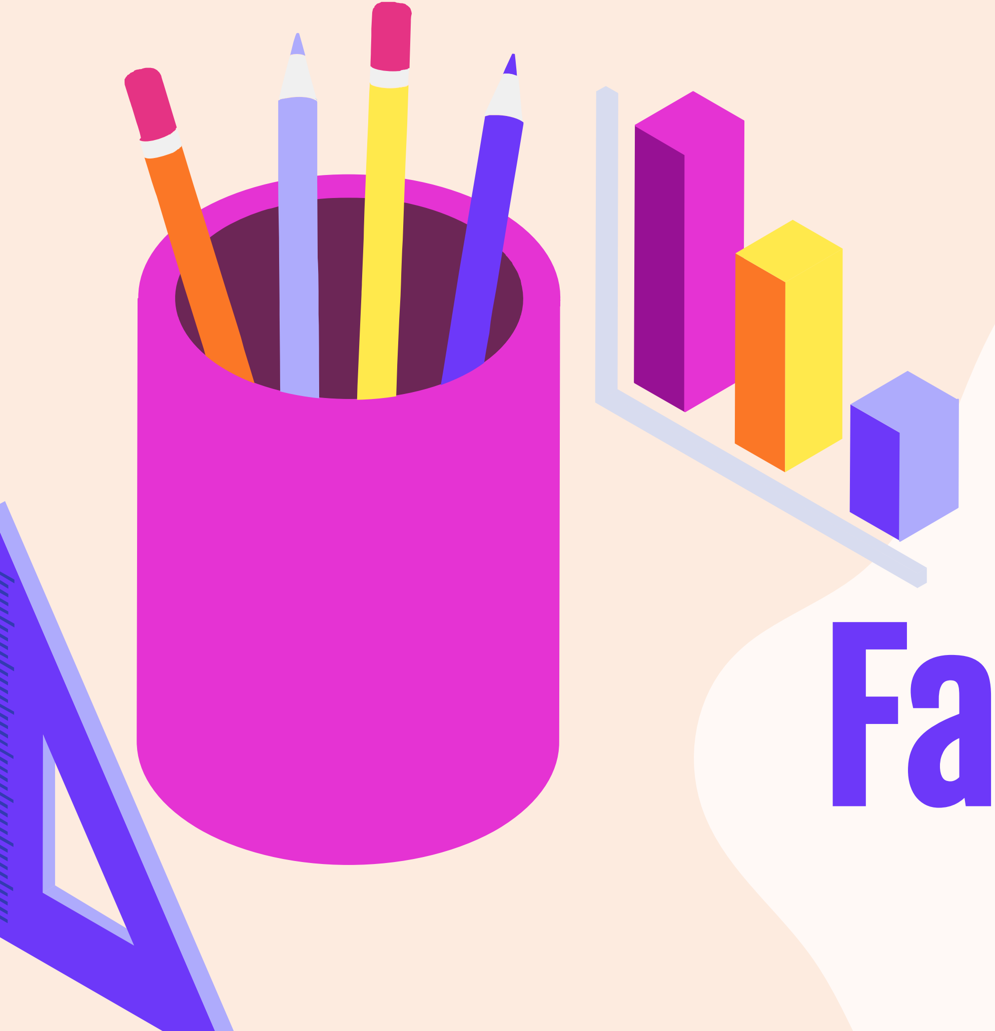
If n = 9, 3 will be printed twice.

Practice

https://codeforces.com/contest/762/problem/A

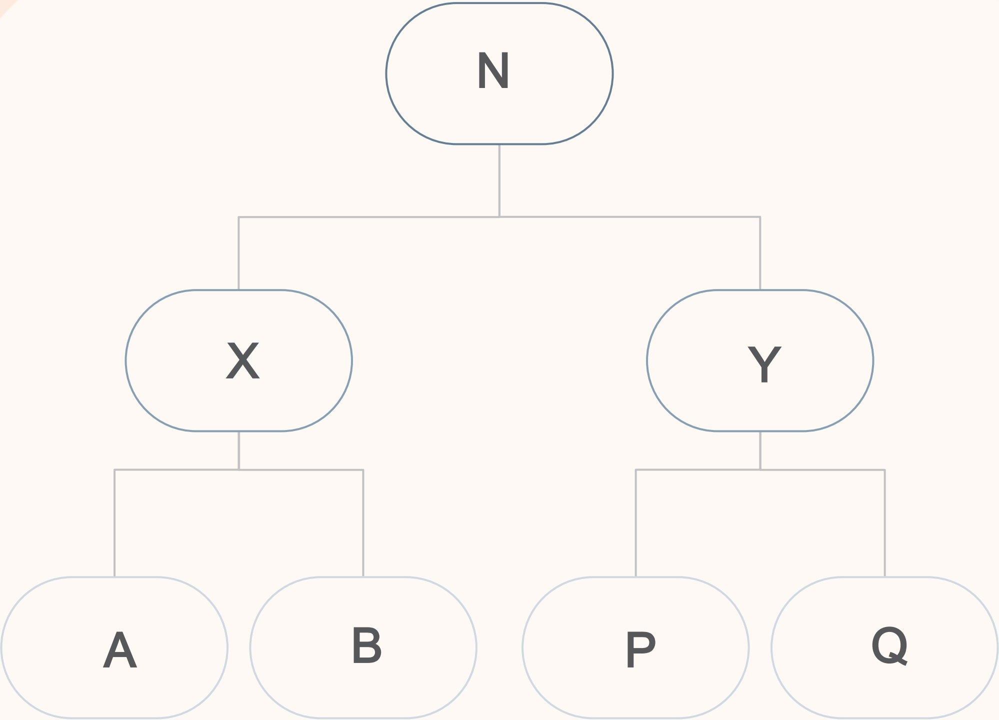
Solution:

https://ideone.com/mlHxlx

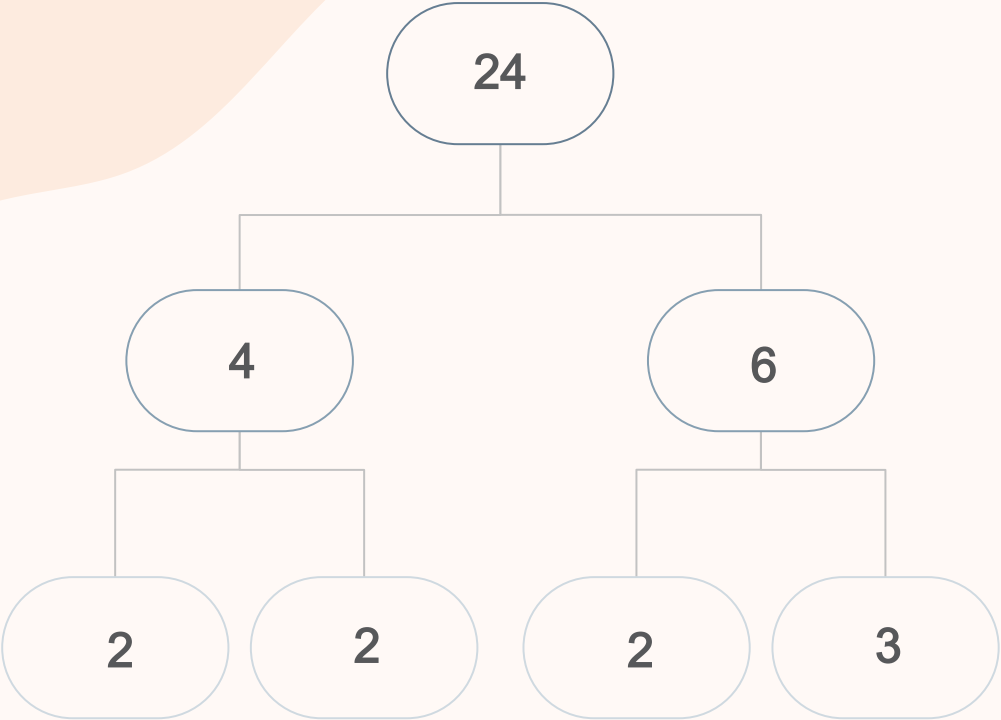


Prime Factorization

Prime Factorization



Prime Factorization

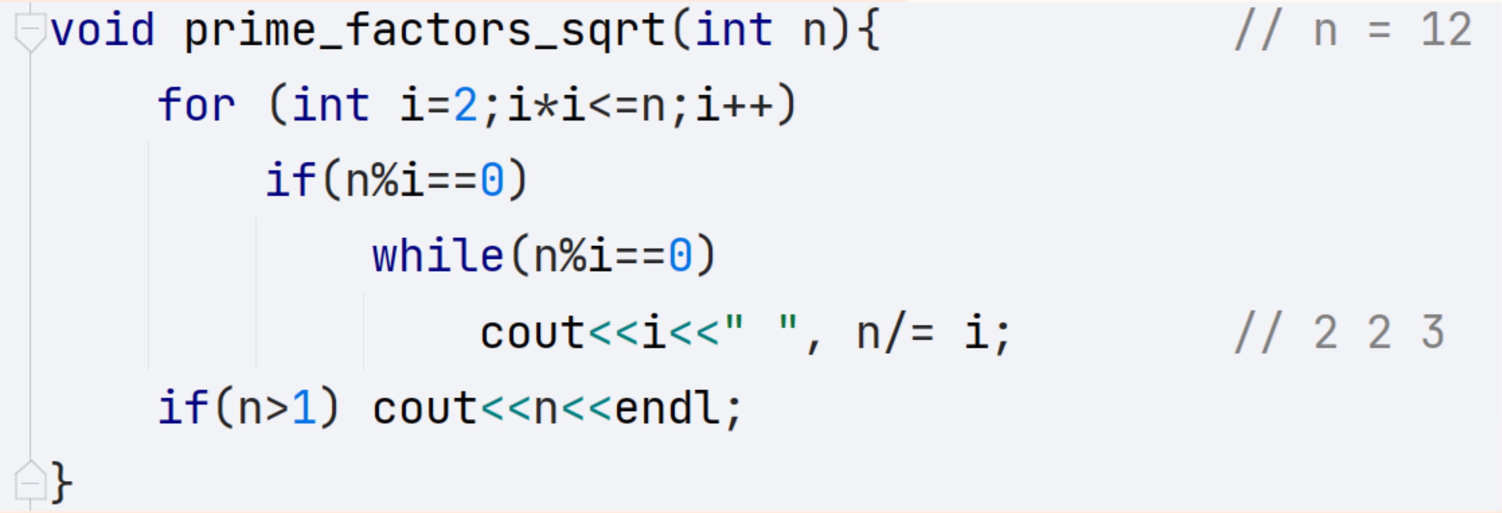


$$N = 2^3 \times 3^1 = 24$$

Division Algorithm

$$N = P_1^a \times P_2^b \times P_3^c \times \dots$$

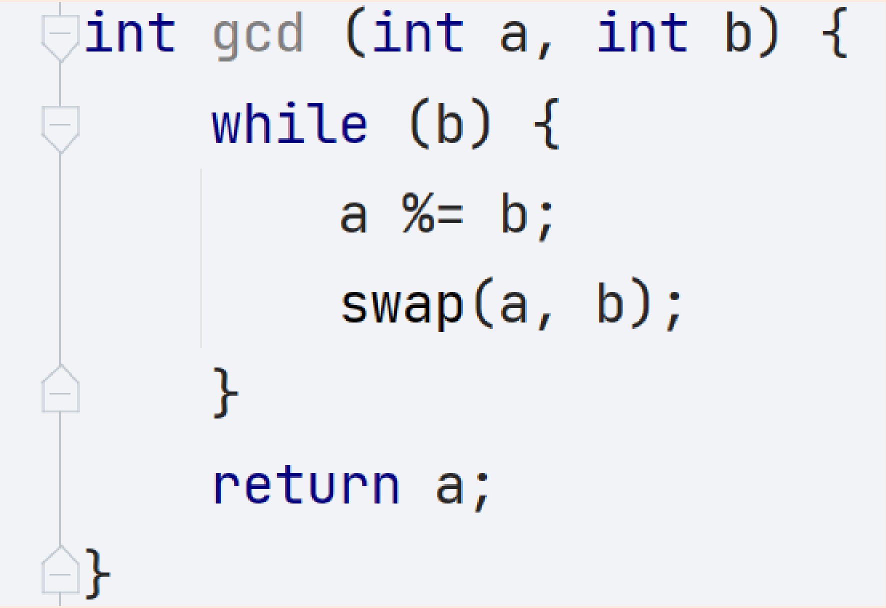
N is the product of all of its prime numbers raised to a certain power



```
void prime_factors_sqrt(int n){                    // n = 12
    for (int i=2;i*i<=n;i++)
        if(n%i==0)
            while(n%i==0)
                cout<<i<<" ", n/= i;               // 2 2 3
    if(n>1) cout<<n<<endl;
}
```

Greatest Common Divisor (GCD)

We can get all divisors and easily take all common divisors and multiply them together. Or…



```
int gcd (int a, int b) {
    while (b) {
        a %= b;
        swap(a, b);
    }
    return a;
}
```

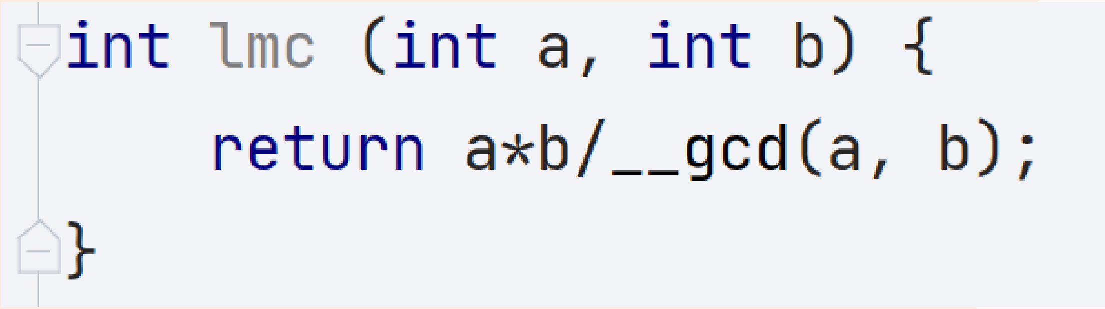
Use __gcd(a, b); built-in function.

Depending on your compiler it could be gcd(a, b); too

O(log(n))

Least/Lowest Common Multiple (LCM)

A method to find the smallest possible multiple of two or more numbers.



```
int lmc (int a, int b) {
    return a*b/__gcd(a, b);
}
```

Better return a/__gcd(a, b)*b; to avoid overflow.

But its still preferable to use long long and focus on constrains.

Practice

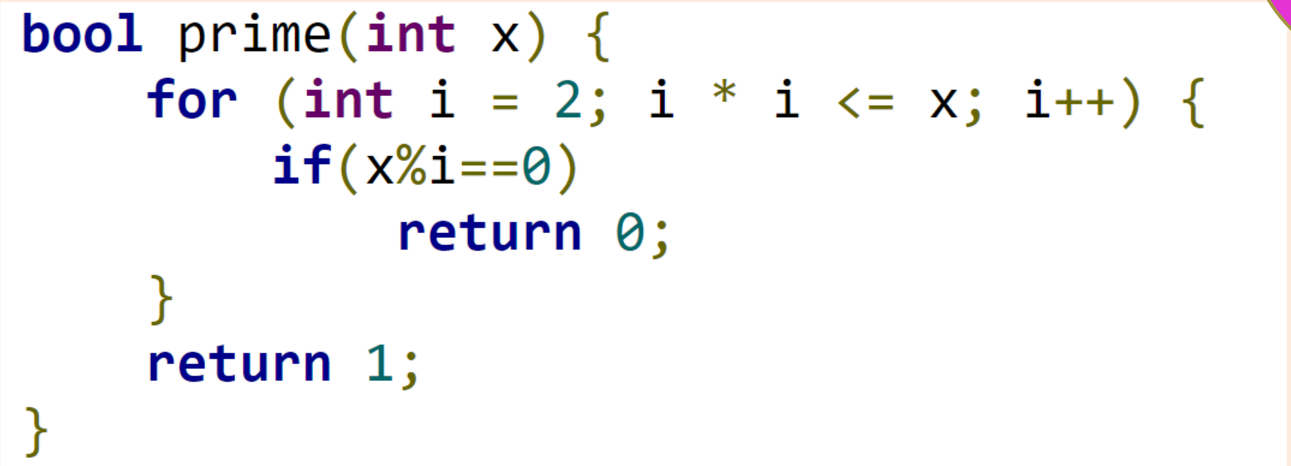
https://codeforces.com/contest/1764/problem/B
Solution: **https://ideone.com/PQhnLD**

PRIME NUMBERS

What if I’m having m queries?

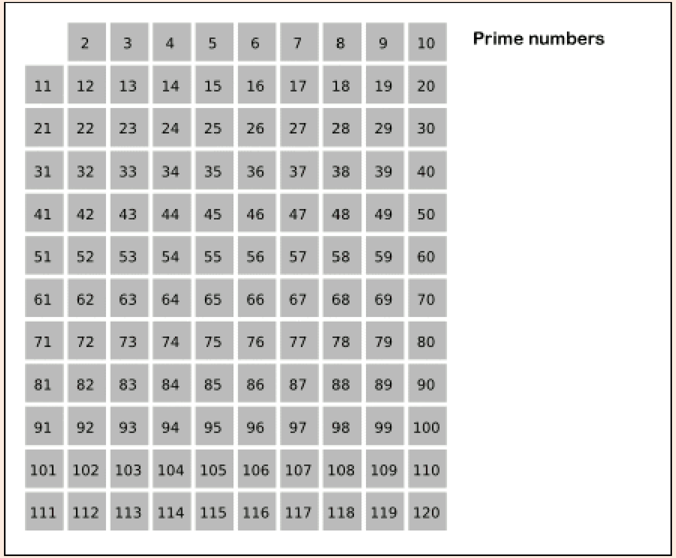
O(m*sqrt(n))…

How can we get it?



```
bool prime(int x) {
    for (int i = 2; i * i <= x; i++) {
        if(x%i==0)
            return 0;
    }
    return 1;
}
```

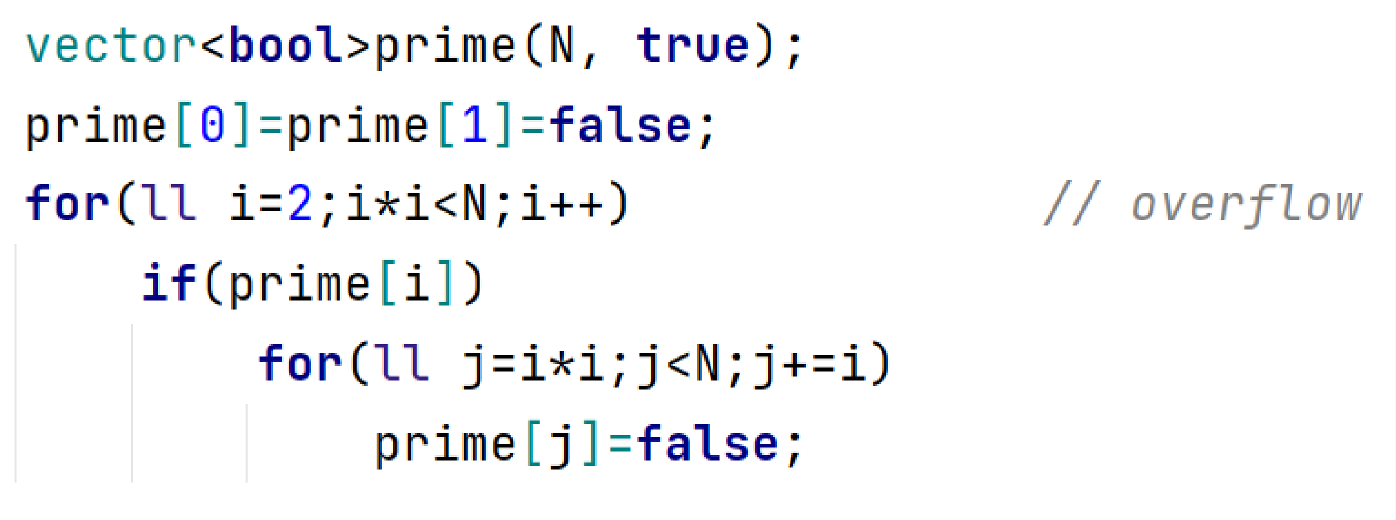
SIEVE OF ERATOSTHENES



	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120

Prime numbers

SIEVE CODE



```
vector<bool>prime(N, true);
prime[0]=prime[1]=false;
for(ll i=2;i*i<N;i++)              // overflow
    if(prime[i])
        for(ll j=i*i;j<N;j+=i)
            prime[j]=false;
```

Time complexit:

n*log(log(n))

Problem:
https://codeforces.com/contest/1474/problem/B
Solution: https://ideone.com/rFpOoN

Resources

https://www.youtube.com/watch?v=gN-nIXpl2rQ&t=2073s
(first 2 hours, third hour is off-topic)

Factorization:
https://www.youtube.com/watch?v=PTxi1Uh6tks
(1:04:40 -> 1:57:30)

Sieve of Eratosthenes:
https://www.youtube.com/watch?v=fuEZwSLmi7g
https://www.youtube.com/watch?v=4ZsieTY8rJ8&t
https://www.youtube.com/watch?v=AyLP2h7qvcE

Thanks

Do you have any questions?

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**