03 May 2024 21:33

Number of factors

$a \% b = 0$

1) a is a factor of 'b'

~~2)~~ b is a factor of 'a'

$10 \% 5 \Rightarrow 0$

1) Initialize a counter = 0.
2) loop from 1 to N. (i)
   a) If $N \% i == 0$
      counter++.
3) return counter.

$a \times b = y$

$x \times x = y$

$b > x$

$a < x$

1, 2, 4, 8, 16

16

1 x 16

2 x 8

4 x 4

18

1 x 18

2 x 9

3 x 6

1, 2, 3, 6, 9, 18

15

1 x 15

3 x 5

$\sqrt{18} \rightarrow 4.$ something

* factors appear in pair

16

1 → 4

Count factors (n).

1) Counter = 0

16	1 - 4	counter
	2	0
1 x 16	3	2
2 x 8	4	4
(4 x 4)		6

1) Counter = 0

2) loop from $(1 - \sqrt{n})$. (i)

a) N % i == 0

counter += 2

2×9

4×4

$O(n) \rightarrow O(\sqrt{n})$

18

$4 \times 4! = 18$

3) If no. is a perfect square counter - - .

$O(n/2) \rightarrow O(n)$

Prime factorization

$16 \rightarrow 2 \times 2 \times 2 \times 2$

$15 \rightarrow 3 \times 5$

$4 \rightarrow 2 \times 2$

Is prime factorization unique? Yes

How to check if a number is prime?

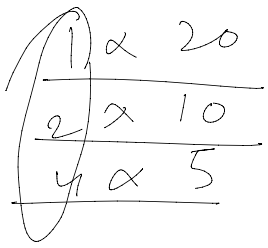
A prime number has exactly 2 factors : 1 and N.

* For every pair of factor, one of the number will be less than or equal to $\sqrt{N}$.

20

4 . something

(20)



1) loop from 2 to N-1 (i):
   a) if (N % i == 0)
      return false;

2) return true;

II (IsPrime)

1) loop from 2 to $\sqrt{N}$ (i):
   a) if (N % i == 0) {
      return false

2) return true.

1) loop from 2 to N (i):
   a) if (N % i == 0 && IsPrime(i)) {
      print(i); N = N / i;

print(i); N = N/i;

i--;

27

3

i	N
2	27
3	9
3	3
	1

27 → 3 × 3 × 3

3 × 3 × 3

```
for (int i=2; i<=N; i++) {
    if( N%i==0 && isprime(i)) {
        N = N/i;
        cout << i << " ";
        i--;
```

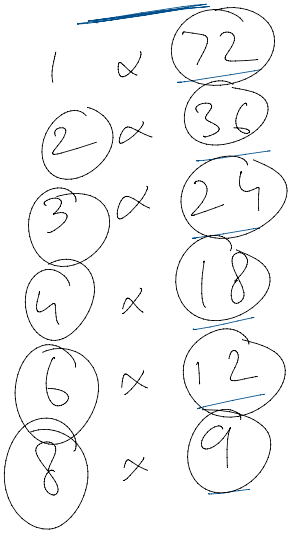
108

i	N
	108
2	54
2	27
2	9
3	3
3	1

2 2 3 3 3

```
int num = 2;
while( num <= N) {
    if (N % num == 0) {
        cout << num << " ";
        N = N / num;
    }
    else {
        num++;
    }
}
```

72



$2 \times 2 \times 2$

3×3

9

1

Prime Number

(A) → return all prime numbers from 1 to A.

A = 10

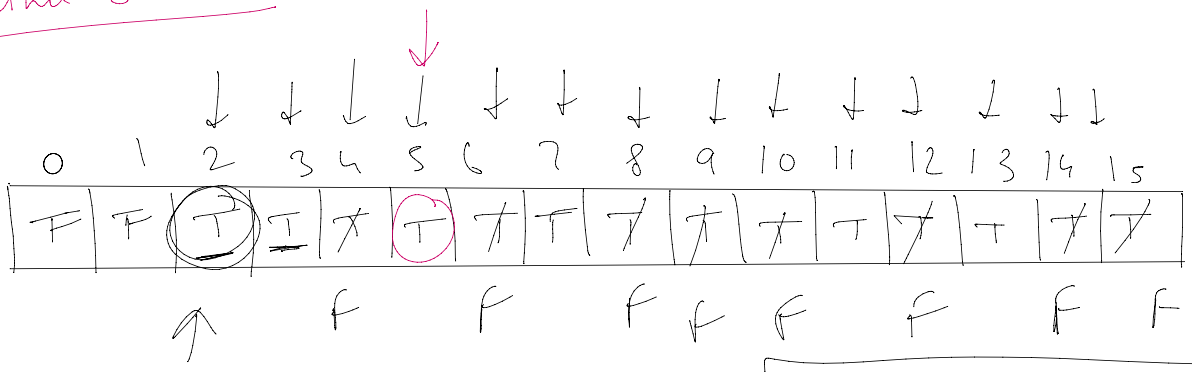
[2, 3, 5, 7]

[2, 3, 5, 7]

1) Loop from 2 to A (i)
   a) check if IsPrime(i)
      i) Include in the ans.

return ans.

Prime Sieve



A = 15

2, 3, 5, 7, 11, 13

5 × 2, 5 × 3, 5 × 5, ...

7 × 2, 7 × 3, 7 × 5, 7 × 6, 7 × 7

1) Declare a boolean array of size A+1 with value

1) Declare a boolean array of size A+1 with values initialized to true (S).

2) Make S[0] = f and S[1] = f.

→ for (int i=2; i<=A; i++) {

if (S[i] == T) {

include i to the answer.

→ for (int j = i; j*i <= A; j++) {

S[i*j] = false;

}

}

3) Return ans

10

i

2 → 4, 6, 8, 10

3 → 9

4

5 → 25

7 → 49

Number of factors for multiple queries.

4

2, 10, 16, 4

⇒ (2, 4, 5, 3)

2 → 1, 2

10 → 1, 10, 2, 5

16 → 1, 2, 4, 8, 16

4 → 1, 2, 4

Q

Time = $O(Q\sqrt{n})$

for (int i = 0; i < Q; i++) {

```
for(int i=0; i<Q; i++){
    int num;
    cin>>num;
    print(countfactors(num));
```

$\sqrt{N}$

3

If we have prime factorization of a number, can we find its' number of factors?

$num \rightarrow x_1^a \times x_2^b \times x_3^c \cdots$

Prime numbers...

$72 \Rightarrow (a+1)(b+1)(c+1)\cdots$

72 → 12

1 × 72

2 × 36

3 × 24

4 × 18

6 × 12

8 × 9

$2 \times 2 \times 2 \times 3 \times 3$

$4 \times 3 = 12$

$2^3 \times 3^2$

2: 0-3

3: 0-2

$2 \times 2 \times 2 \times 3$

2

2×2

$2 \times 2 \times 2$

$2 \times 2 \times 3$

3×3

---

spf : smallest prime factor

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
-1	-1	2	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	0	ϕ
		3	2	5	2	7	2	3	2		11	2	13	2	3	2	17	2

13 14

3 × 5

2 × 7

1) Create an integer array of size A+1, initialized to 0.

```
for(int i=2; i<=A; i++) {
    if(spf[i]==0) {
        spf[i]=i;
        for(int j=i; j*i<=A; j++) {
            if(spf[j*i]==0) {
                spf[j*i]=i;
            }
        }
    }
}
```

Prime

> optimally find prime numbers.

> Prime factorization

> Seive

> Seine
> SPF