* **Prime Number** : No. which has exactly 2 factors [1 & No. itself]

⇒ Is 1 a prime No ? X

⇒ **Composite No.** : No. of factors > 2

⇒ 1 is neither prime nor composite.

⇒ Count of factors.

→ O(N)

→ O($\sqrt{N}$)

⇒ factors appears in a pair

$a * b = N$

a & b are factors of N

a & $\frac{N}{a}$ are factors of N

N = 24

i	N/i	Count = 0
1	24	+2
2	12	+2
3	8	+2
4	6	+2
6	4	
8	3	
12	2	
24	1	

N = 100

i	N/i
1	100
2	50
4	25
5	20
10	10
20	5
25	4
50	2
100	1

Count = 0
+2
+2
+2
+2
+1

$i <= \frac{N}{i}$

$i * i <= N$

$i <= \sqrt{N}$

```
Count = 0                          Sqrt(N) { i*i <= N }
for( i = 1; i <= √N ; i++ ){
    if ( N % i == 0 ){
        if(i == N/i) count += 1
        else  count += 2
    }
}
```

TC: $O(\sqrt{N})$
SC: $O(1)$

if (count == 2) ⇒ N is a prime no.
else N isn't a prime no.

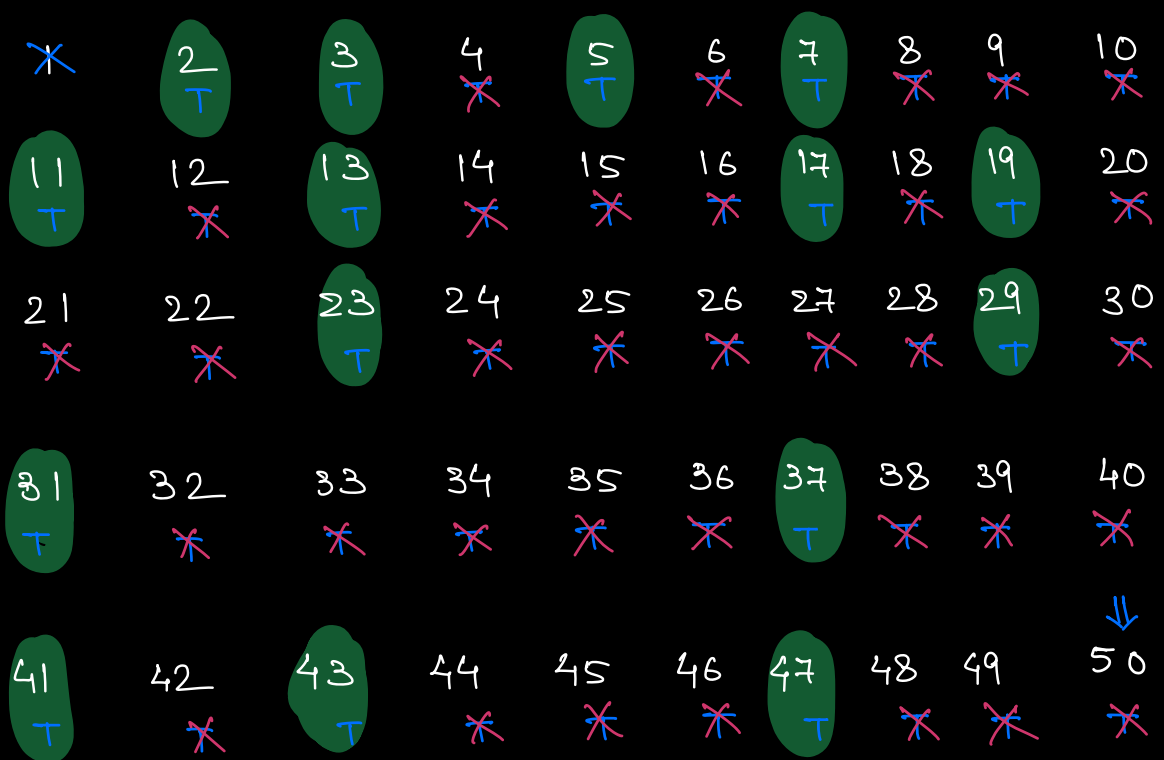
Q. Given a number N, print all the prime no's from 2 to N.

N = 10 ⇒ 2, 3, 5, 7

```
for(i=2; i<=N; i++){
    if( isPrime(i) ){        → O(√N)
        print(i)
    }
}
```

TC : O(N · sqrt(N))

⇒ N = 50



→ Assuming all the no's to be prime initially.

→ (2) cancels out its multiples.

(3) _____

Steps

1) bool isPrime[N+1] = {true};

isPrime[0] = false

isPrime[1] = false

2) Go from i = 2 to N

& if (i) is a prime no then mark all the multiples of i to be false.

```
bool isPrime[N+1] = {true};
isPrime[0] = false
isPrime[1] = false
for(i=2; i<=N; i++){
    if(isPrime[i]){
        for(j=2*i; j<=N; j+=i){
            isPrime[j] = false;
        }
    }
}
```

Only runs when i is prime

```
for( i= 2; i<= N; i++ ){
    if( isPrime[i] ) print(i);
}
```

Multiples of 2 in $[1, N] \Rightarrow \frac{N}{2}$

Multiples of 3 in $[1, N] \Rightarrow \frac{N}{3}$

$$\# \text{ of iterations} = \frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \frac{N}{7} + \cdots\cdots$$

$$= N\left(\underbrace{\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \cdots\cdots}_{\text{Sum of reciprocal of all the prime no's from } [2,N]}\right)$$

$$\left(\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \cdots\cdots\right) < \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} \cdots \frac{1}{N}$$

$$\int_{x=2}^{N} \frac{1}{x}\,dx \rightarrow \log_2 N$$

Upper Bound of TC : $N \log_2 N$

Actual TC :-

$$O(N \log(\log N))$$

$N = 2^{64}$

$\hookrightarrow \log N = 64$

$\hookrightarrow \log \log N = \log 64 = 6$

Observations.

$\Rightarrow$ First multiple cancelled by 2 $\Rightarrow 2 \times 2$

$\Rightarrow$ " " " " 3 $\Rightarrow 2 \times 3$

$\Rightarrow$ " " " " 5 $\Rightarrow 2 \times 5$

⋮

" " " " i $\Rightarrow 2 \times i$

$5 \Rightarrow 2\times5,\ 3\times5,\ 4\times5,\ 5\times5,\ \dots$

$\pi \Rightarrow 2\times\pi,\ 3\times\pi,\ 4\times\pi,\ 5\times\pi,\ 6\times\pi,\ \pi\times\pi,\ \dots$

$i \Rightarrow 2i,\ 3i,\ 4i,\ 5i,\ \dots,\ i\times i$

```
bool isPrime[N+1] = {true};
isPrime[0] = false
isPrime[1] = false
for(i=2; i*i<=N; i++){
    if(isPrime[i]){
        for(j=i*i; j<=N; j+=i){
            isPrime[j] = false;
        }
    }
}
```

Only runs when i is prime

✗	2 T	3 T	4 ✗	5 T	6 ✗	7 T	8 ✗	9 ✗	10 ✗
11 T	12 ✗	13 T	14 ✗	15 ✗	16 ✗	17 T	18 ✗	19 T	20 ✗
21 ✗	22 ✗	23 T	24 ✗	25 ✗	26 ✗	27 ✗	28 ✗	29 T	30 ✗
31 T	32 ✗	33 ✗	34 ✗	35 ✗	36 ✗	37 T	38 ✗	39 ✗	40 ✗
41 T	42 ✗	43 T	44 ✗	45 ✗	46 ✗	47 T	48 ✗	49 ✗	50 ✗

TC :- $O(N \log(\log N))$

$\log(\log N) << \log N << N$

Sieve of Eratosthenes.

Q: Given a no. N, find the smallest prime factor (SPF) for all no's from 2 to N.

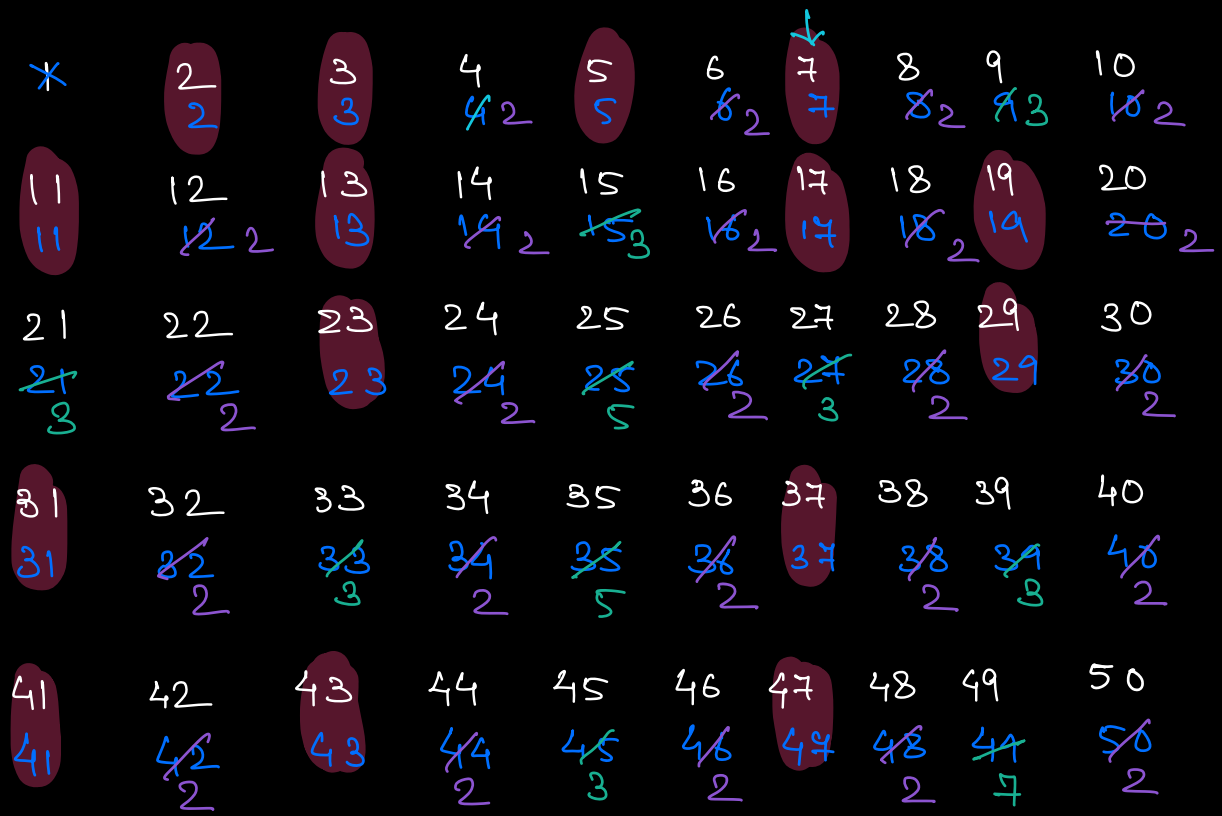
SPF(10) = 2
SPF(2) = 2
SPF(15) = 3
SPF(28) = 2
SPF(21) = 3
SPF(25) = 5
SPF(11) = 11

SPF of a Prime will be number itself.

N = 10

	2	3	4	5	6	7	8	9	10
spf[]:	2	3	2	5	2	7	2	3	2

```
int spf[N+1];
spf[i] = i  (i ∈ [2, N])
```



TC: $O(N \log \log N)$

⇒ Code

```
int spf[N+1];
for(i=2; i<=N; i++)
    spf[i] = i;
for(i=2; i<=√N; i++){
    if(spf[i] == i){
        // i is a prime no.
        // Mark all the multiples of i as false
        for(j=i*i; j<=N; j+=i){
            spf[j] = min(spf[j], i);
        }
    }
}
```

OR

```
if(spf[j] == j)
    spf[j] = i;
```

Divisors

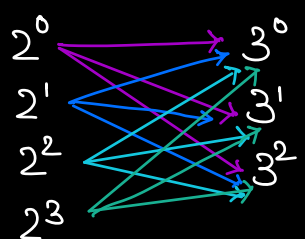
⇒ Count the no. divisors for N = 72

$$
\begin{array}{r|l}
2 & 72 \\
\hline
2 & 36 \\
\hline
2 & 18 \\
\hline
3 & 9 \\
\hline
3 & 3 \\
\hline
 & 1
\end{array}
$$

Prime factorisation of $72 = 2 \times 2 \times 2 \times 3 \times 3$

$= 2^3 \times 3^2$

$\{2^0, 2^1, 2^2, 2^3\}$ $\{3^0, 3^1, 3^2\}$



⇒ # of divisors $= 4 \times 3$

$= 12.$

$2^0 * 3^0$	$2^1 \times 3^0$	$2^2 \cdot 3^0$	$2^3 \cdot 3^0$
$2^0 \times 3^1$	$2^1 \times 3^1$	$2^2 \cdot 3^1$	$2^3 \cdot 3^1$
$2^0 \times 3^2$	$2^1 \times 3^2$	$2^2 \cdot 3^2$	$2^3 \cdot 3^2$

$N = 600$

spf(600) ←

2	600
2	300
2	150
3	75
5	25
5	5
	1

Prime factorisation =

$2^3 \cdot 3^1 \cdot 5^2$

{4} {2} {3}

⇒ # of divisors = $4 \times 2 \times 3$

$= \underline{24}$

General :

$$N = p_1^{x_1} \times p_2^{x_2} \times p_3^{x_3} \times \ldots\ldots \times p_n^{x_n}$$

# of divisors ⇒ $(x_1+1)(x_2+1)(x_3+1) \ldots\ldots (x_n+1)$

```
// Build spf[]
divisors = 1
while (N > 1) {
    x = spf[N]
    c = 0
    while(N % x == 0) {
        c++
        N = N/x
    }
    divisors *= (c+1);
}
```

N = 72 , ans = 1

x = spf[72] = 2

c = 0 1 2 3

N = 36 18 9

ans *= (c+1)

ans = 4

N = 9

x = spf[9] = 3

c = 0 1 2

N = 3 1

ans * (c+1)

4 * (3) = 12

TC: $O(N \log \log N) + O(\log N)$

SPF — Count of divisors.

$: O(N \log \log N)$

SC: $O(N)$

↳ SPF[]

* Prime factors of N

2	360
2	180
2	90
3	45
3	15
5	5
	1

$360 \Rightarrow 2^3 \cdot 3^2 \cdot 5^1$

primefactors:

2	2	2	3	3	5

```
// Build Spf []
List<int> primefactors;
while (N > 1) {
    x = spf[N]
    N = N/x
    primefactors.add(x);
}
```

N = 360

$x = 2$

$N = \frac{360}{2} = 180$

$x = 2, N = \frac{180}{2} = 90$

$x = 2, N = \frac{90}{2} = 45$

$x = 3, N = \frac{45}{3} = 15$

⋮

2, 2, 2, 3 ....

Q: Given N, for every no. 1 to N, get the no. of factors.

N = 10	1	2	3	4	5	6	7	8	9	10
factors[] :	1	2	2	3	2	4	2	4	3	4

```
for( i=2; i<=N; i++){
    // Above Code
}
```

—— * ——