Prime Number :- No. which has exactly (2) factors
i.e (1) & Number itself.

(1) ⇒ X Not prime

(2) ⇒ only even prime no.

Count of factors ⇒ O(N)
↳ O(sqrt(N))

N=24

i	N/i
1	24
2	12
3	8
4	6
6	4
8	3
12	2
24	1

N= 100

i	N/i
1	100
2	50
4	25
5	20
$\boxed{10}$	$\boxed{10}$
20	5
25	4
50	2
100	1

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

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

$$i*i <= N$$

$$\boxed{i <= sqrt(N)}$$

* factors appears in pairs.

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

if (count==2) : Prime No.

else : Not a prime no.

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

```
# bool isPrime (N) {        N>1
      for( i=2; i*i <=N; i++){
            if (N % i ==0)
                  return false;
      }
      return true;
}
```

Q.1 Given N, print all the prime Numbers from 1 to N.

$N = 10 \Rightarrow 2, 3, 5, 7$

Brute force

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

$O(\sqrt{N})$

TC: O(Nsqrt(N)) ⇒ upper bound of TC

SC: O(1)

N = 50

1	2 T	3 T	4 ~~T~~ F	5 T	6 ~~T~~ F	7 T	8 ~~T~~ F	9 ~~T~~ F	10 ~~T~~ F
11 T	12 ~~T~~ F	13 T	14 ~~T~~ F	15 ~~T~~ F	16 ~~T~~ F	17 T	18 ~~T~~ F	19 T	20 ~~T~~ F
21 ~~T~~ F	22 ~~T~~ F	23 T	24 ~~T~~ F	25 ~~T~~ F	26 ~~T~~ F	27 ~~T~~ F	28 ~~T~~ F	29 T	30 ~~T~~ F
31 T	32 ~~T~~ F	33 ~~T~~ F	34 ~~T~~ F	35 ~~T~~ F	36 ~~T~~ F	37 T	38 ~~T~~ F	39 ~~T~~ F	40 ~~T~~ F
41 T	42 ~~T~~ F	43 T	44 ~~T~~ F	45 ~~T~~ F	46 ~~T~~ F	47 T	48 ~~T~~ F	49 ~~T~~ F	50 ~~T~~ F

Steps :-

1) Create a boolean Array of size N+1

   bool isPrime [N+1] = {True};

2) for every no. from i = 2 to N

   * if isPrime [i] == true ⇒ mark all the multiples of i to false in isPrime Array.

```
bool     Prime[N+1] = {true};
isPrime[0] = false;
isPrime[1] = false;
for(i=2; i<=N; i++){
        if( prime[i]){
            for(j= 2*i; j<=N; j+=i ){
                  Prime[j] = false;
            }
        }
}
// Print Prime No's
for( i=2 ; i<=N; i++){
        if(Prime[i])
            Print(i)
}
```

N log N

O(N)

Time Complexity analysis :-

$$\text{Iterations} = \underbrace{\frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \frac{N}{7} + \frac{N}{11} + \cdots\cdots}_{\text{Sum of reciprocal of prime no's.}}$$

$$= N\left(\frac{1}{2}+\frac{1}{3}+\frac{1}{5}+\frac{1}{7}+\frac{1}{11}+ \cdots\right)$$

$$= N\left(\underbrace{\int_{i=2}^{N}\frac{1}{i}\,di}\right)$$

$$= N * (\log N)$$

TC : O(N log N) {Approximate}

SC : O(N) → prime array

Actual TC :-

$O(N * \log(\log N))$ ⇒ TODO

→ $\simeq O(N)$

$\log N >> \log(\log(N))$

$N = 2^{64} \simeq 10^{18}$

$\log_2 N = 64$

$\log(\log(N)) = \log(64) = 6$

Sieve of Eratosthenes

Observation 1

first multiple marked as false by 2 = 2*2

first multiple marked as false by 3 = 2*3

first multiple marked as false by 5 = 2*5

first multiple marked as false by 7 = 2*7

```
for( i = 2; i <= N; i++ ) {
        if ( prime[i]) {
             for(j = i*i; j<=N; j+=i ) {
                    Prime[j] = false;
             }
        }
}
```

Observation 2

~~1~~	2 T	3 T	4 ~~T~~ F	5 T	6 ~~T~~ F	7 T	8 ~~T~~ F	9 ~~T~~ F	10 ~~T~~ F
11 T	12 ~~T~~ F	13 T	14 ~~T~~ F	15 ~~T~~ F	16 ~~T~~ F	17 T	18 ~~T~~ F	19 T	20 ~~T~~ F
21 ~~T~~ F	22 ~~T~~ F	23 T	24 ~~T~~ F	25 ~~T~~ F	26 ~~T~~ F	27 ~~T~~ F	28 ~~T~~ F	29 T	30 ~~T~~ F
31 T	32 ~~T~~ F	33 ~~T~~ F	34 ~~T~~ F	35 ~~T~~ F	36 ~~T~~ F	37 T	38 ~~T~~ F	39 ~~T~~ F	40 ~~T~~ F
41 T	42 ~~T~~ F	43 T	44 ~~T~~ F	45 ~~T~~ F	46 ~~T~~ F	47 T	48 ~~T~~ F	49 ~~T~~ F	50 ~~T~~ F

```
for( i = 2 ; i*i <= N; i++ ) {
    if (prime[i]) {
        for (j = i*i; j<=N; j+=i ) {
            prime[j] = false;
        }
    }
}
```

$j = i*i$ Last value of $i = \sqrt{N}$

$i = \sqrt{N}+1 \Rightarrow j = (\sqrt{N}+1)(\sqrt{N}+1)$

$j = N + 1 + 2\sqrt{N} > N$

TC : $O(N \cdot \log(\log(N)))$ → **Todo.** }

SC : $O(N)$

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

10 ⇒ SPF = 2
15 ⇒ SPF = 3
23 ⇒ SPF = 23
11 ⇒ SPF = 11

SPF of a prime no. is the no. itself.

N = 10

	2	3	4	5	6	7	8	9	10
SPF	2	3	2	5	2	7	2	3	2

SPF[int]

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

SPF[i] == i ⇒ i is a prime number.

```
int spf[N+1];
for (i=2; i<=N; i++){
    spf[i] = i;
}
```

O(N)

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

O(NloglogN))

⇒ spf[j] = min(spf[j], i);

TC: O(N log(log(N)))

SC: O(N) ↳ SPF Array

Divisors :-

$N = 72$

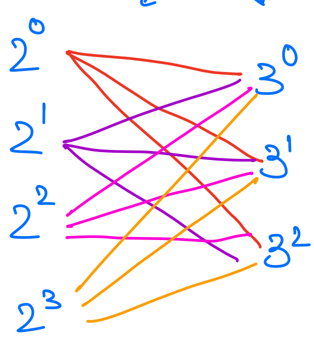
2	72
2	36
2	18
3	9
3	3
	1

$N = 72 = 2 \times 2 \times 2 \times 3 \times 3$

$= 2^3 \cdot 3^2$

Prime factorisation

$N = 72 = 2^3 \cdot 3^2$



$2^0 3^0$	$2^1 3^0$	$2^2 3^0$	$2^3 3^0$
$2^0 3^1$	$2^1 3^1$	$2^2 3^1$	$2^3 3^1$
$2^0 3^2$	$2^1 3^2$	$2^2 3^2$	$2^3 3^2$

$\Downarrow$

All the divisors of $N=72. \Rightarrow \boxed{12}$

$N = 72 = 2^{3} 3^{2}$

Count of divisors $= (3+1)(2+1)$

$= 4 \cdot 3$

$= 12$ *

$N = 600$

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

$600 = 2 \times 2 \times 2 \times 3 \times 5 \times 5$

$= 2^3 3^1 5^2$

# of divisors $= (3+1)(1+1)(2+1)$

$= 4 \cdot 2 \cdot 3$

$= 24$

$2^3 \Rightarrow 2^0, 2^1, 2^2, 2^3$
$3^1 \Rightarrow 3^0, 3^1$
$5^2 \Rightarrow 5^0, 5^1, 5^2$

Generalize

$$N = P_1^{x_1} \times P_2^{x_2} \times P_3^{x_3} \cdots\cdots \times P_n^{x_n}$$

$$\# \text{ of divisors} = (x_1+1)(x_2+1)(x_3+1)\cdots(x_n+1)$$

Steps

1) Create SPF $\Rightarrow O(N \log(\log N))$

2)
```
divisors = 1
while (N > 1) {
    x = SPF[N]
    power = 0
    while (N % x == 0) {
        N = N / x
        power++;
    }
    divisors = divisor * (power + 1);
}
return divisors.
```

$O(\log_2 N)$

upper bound.

$N = \underline{\underline{600}} = 2^3$

$ans = 1 * (4) = 4$

$ans = 4 * (2) = \underline{\underline{8}}$

$ans = 8 * (2+1) = \underline{\underline{24}}$

$x = 5$

$c = \cancel{0}\ \cancel{1}\ 2$

$N = \cancel{600}\ \cancel{300}\ \cancel{150}\ \cancel{75}\ \cancel{25}\ \cancel{5}\ 1$

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

SC: $O(N)$ → for SPF Array

Prime factors of N

$N = 72 = 2^3 \cdot 3^2$

Prime factors of 72:

2	2	2	3	3

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

Q. Given N, for every number [1 to N].
Get the no. of factors.

N = 10

1	2	3	4	5	6	7	8	9	10
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
1	2	2	3	2	4	2	4	3	4

```
factors[1] = 1
for( i=2 ; i<=N ; i++){
    // Above code
}
```

Doubts

$$2^{10} = 1024 \approx 1000 = 10^3$$

$$2^{60} = 2^{10} \cdot 2^{10} \cdot 2^{10} \cdot 2^{10} \cdot 2^{10} \cdot 2^{10}$$

$$\approx 10^{18}$$