Count of factors of N → TC = $O(\sqrt{N})$

SC = $O(1)$

```
f = 0
for i → 1 to √N
    if (N % i == 0)
        if (i == N/i)
            f += 1
        else
            f += 2
return f
```

Prime Number >0

Number having exactly 2 factors (+ve/Natural) 1 & itself.

2, 3, 5, 7, 11, 13, 17, 19 . . .

↑ smallest prime number

Q→ Given an integer N. Check every number from 1 to N if it is a prime number.

N = 10

	1	2	3	4	5	6	7	8	9	10
isP →	F	T	T	F	T	F	T	F	F	F

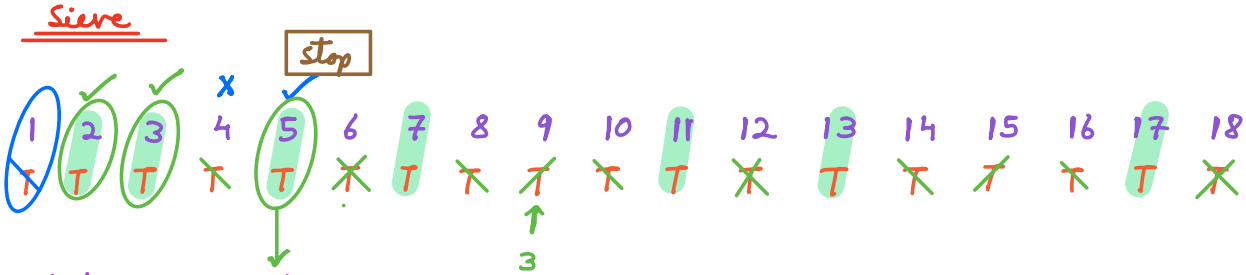
(Ans)

Bruteforce → ∀ no. 1 to N, count #factors if it is = 2
current number is prime
else current number is not prime.

TC = $O(N\sqrt{N})$

SC = $O(1)$

Sieve



what is the first no. that 5 can remove → 10 → 2 * 5, 15 → 3 * 5, 20 → 2 * 2 * 5, 25 → 5 * 5

$x \rightarrow$ first multiple to be removed $= x^2$

$\forall i$, isP[i] = true
isP[0] = isP[1] = false

i = 3
9 + 3 = 12
12 + 3 = 15
15 + 3 = 18

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

SC = O(1) isP → o/p

Time Complexity

i	j	# iterations
2	4, 6, 8, 10 ...	N/2
3	9, 12, 15, ...	= N/3
4	———	0
5	25, 30, 35 ...	≃ N/5
⋮		⋮

$$\frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \frac{N}{7} + \frac{N}{11} \cdots$$

$$N\left(\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \frac{1}{11} \cdots\right) \quad <= \quad N\left(\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} \cdots\right)$$

sum of reciprocal of prime → $\log(\log(N))$

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

sum of reciprocal of all numbers

$$\sum_{x=1}^{N} \frac{1}{x} \simeq \int \frac{1}{x}\, dx = \log(x)$$

$N * \log(N) \rightarrow TC \leq O(N \log(N))$ ✓

$$\log_2(\log_2(2^{32})) = \log_2(32) = \log_2(2^5) = 5$$

$\log_2(2^x) = x$

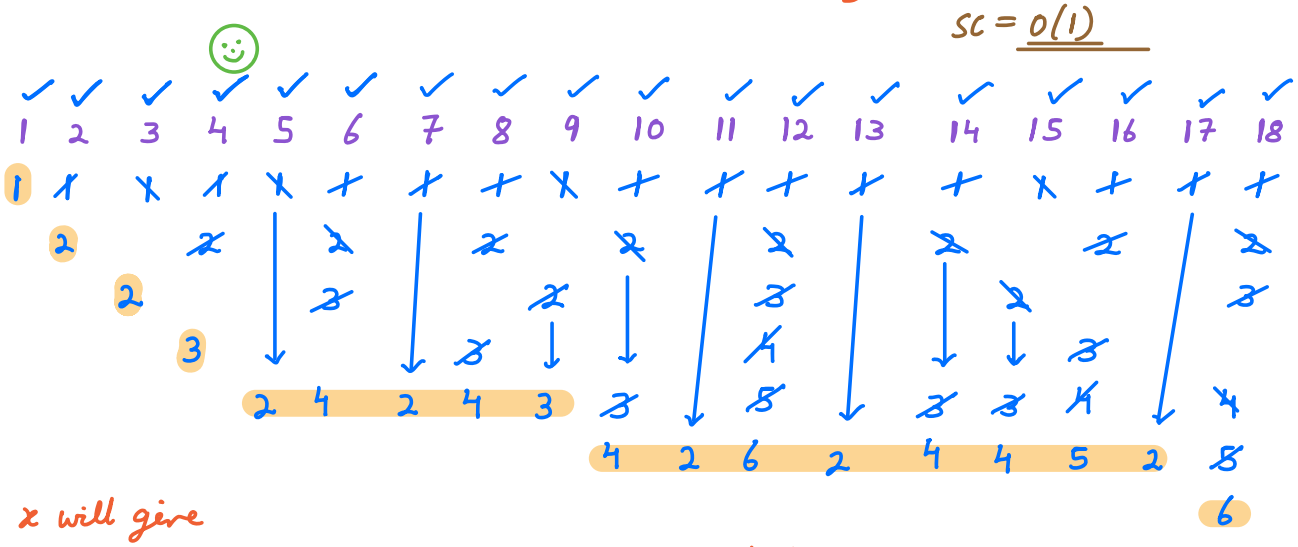
$\log_a(b) = c \Rightarrow a^c = b$

Q→ Given an integer N.
For all numbers from 1 to N count the factors of the number.

N = 8

	1	2	3	4	5	6	7	8	
factors [	1	2	2	3	2	4	2	4]	(Ans)

Bruteforce → ∀no. count factors separately → TC = $O(N\sqrt{N})$
SC = O(1)



x will give
first chocolate to → x then to other multiples of x.

```
∀i, cnf[i] = 0
for (i = 1 ; i <= N ; i++) {
    for (j = i ; j <= N ; j += i) {
        cnf[j] += 1
    }
}
return cnf
```

SC = O(1) cnf[] → o/p
TC = O(N log(N)) ✓

i	j	# iterations
1	1, 2, ... N	N
2	2, 4, 8 ...	N/2
3	3, 6, 9 ...	N/3
4	4, 8, 12 ...	N/4
		↓

$N * \left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} \ldots\right)$

i/p → Algorithm → o/p
N ✓ spf[]

10:35 PM

Q → Given an integer N.
Find the smallest prime factor for all numbers from 1 to N.
1 → spf = 0 (SPF)

N = 10

1	2	3	4	5	6	7	8	9	10
spf → 0	2	3	2	5	2	7	2	3	2] (Ans)

✓ ✓ ☹ ✓

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

25 stop

x → start from = x^2

1	2	3	4	5
0	2	3	2	5
6	7	8	9	10
2	7	2	3	2
11	12	13	14	15
11	2	13	2	3

```
∀i, spf[i] = 0
for (i = 2 ; i*i <= N ; i++) {
    if (spf[i] == 0) {  // i → prime
        spf[i] = i
        for (j = i*i ; j <= N ; j += i) {
            if (spf[j] == 0)
                spf[j] = i
        }
    }
}
for (i = 2 ; i <= N ; i++) {
    if (spf[i] == 0)
        spf[i] = i
}
```

TC = $O(N \log(\log(N)) + N)$
$\simeq O(N \log(\log(N)))$

SC = $O(1)$ spf → o/p

20 → spf[20] < 20 if 20 is not prime

1 * 20

2 * 10
4 * 5 → spf[20] > 0 before we reach 20 (i = 20).

Count number of factors of N given the spf[i] → ∀i [1 to N]

45 → 6 factors 1 3 5 9 15 45

32 → 6 factors 1 2 4 8 16 32
2^5 → 2^0 2^1 2^2 2^3 2^4 2^5 $2^0 \rightarrow 2^5$

factors of 2^K = $\{2^0, 2^1, 2^2 \ldots 2^K\}$ → (K+1) ✓

45 → 6 factors 1 3 5 9 15 45

$3^2 * 5^1$

3^0*5^0 3^0*5^1 3^1*5^1
3^1*5^0 3^2*5^0 3^2*5^1

3^0 3^1 3^2 5^0 5^1

(2+1) * (1+1)
3 * 2 = 6

factors of $3^x * 5^y$ → (x+1)*(y+1)

factors of $2^a * 3^b * 5^c$ → (a+1)*(b+1)*(c+1) ✓

N = 490 → $2^1 5^1 7^2$ → # factors = (1+1)*(1+1)*(2+1)
= 2*2*3 = 12

spf

490
spf = 2
490 → 245 (÷2) → 2^1
245
spf = 5
245 → 49 (÷5) → 5^1
49
spf = 7
49 → 7 → 1 (÷7) → 7^2

600
spf = 2
600 → 300 → 150 → 75 (÷2) → 2^3
75
spf = 3
75 → 25 (÷3) → 3^1
25
spf = 5
25 → 5 → 1 (÷5) → 5^2

```
f = 1
while (N > 1) {
    s = spf[N]
    pow = 0
```

N → N/spf <= N/2

N/3 < N/2
N/5 < N/2
⋮
N/p <= N/2

p >= 2

```
while (N % s == 0) {
    N /= s
    pow += 1
}
f *= (pow + 1)
}
return f
```

$N \rightarrow N/2 \Rightarrow TC \leq O(\log(N))$ ✓ (multiple queries)

precomputation of spf[] → $TC = O(N \log(\log(N)))$

$SC = O(N)$

multiple queries

$TC \rightarrow O(Q * \log(N) + N \log(\log(N)))$