Maths 3: Prime Numbers

AGENDA

- → Introduction to Prime Numbers
- → Get all primes from 1 to N.
- → Print Smallest prime factors for 2 to N.
- → Prime Factorization
- → Get the no. of factors / divisors

Question :- what are prime Numbers ?

↳ Number having only 2 factors i.e. 1 & the Number itself are known as Prime Number.

Ex :-

1 ⟶ 1 ✗

2 ⟶ 1 2 ✓

5 ⟶ 1 5 ✓

7 ⟶ 1 7 ✓

9 ⟶ 1 3 9 ✗

11 ⟶ 1 11 ✓

Question :- Given a number, we need to check whether its a prime number or not.

Brute force

↳ Count of factors by going through 1 to $\sqrt{N}$.

PSEUDO CODE

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

if ( count == 2 ) {
    return True;
} else {
    return False;
}
```

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

Question :- Given a number N, we need to print all the prime no. from 1 to N.

Ex:- N = 10 → 2, 3, 5, 7

N = 20 → 2, 3, 5, 7, 11, 13, 17, 19

Brute force

↳ Iterate from 1 to N, & check if a no is prime or not.

PSEUDO CODE

```
for (i=2; i<=n; i++){
    if (checkprime(i)){
        Print(i);
    }
}
```

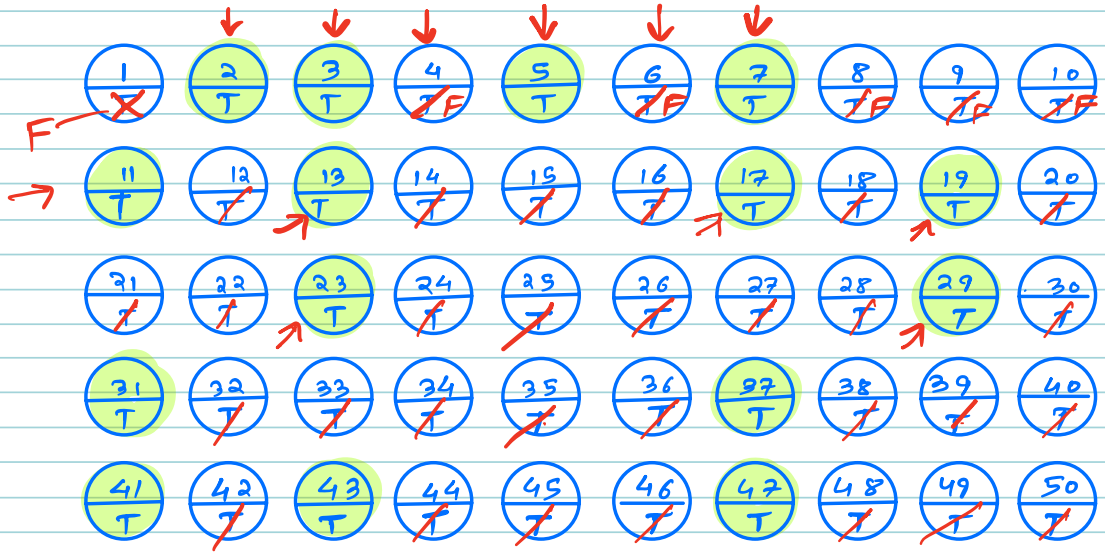
TC — $O(N\sqrt{N})$

SC — $O(1)$

↳ OPTIMIZATION

INITIAL ASSUMPTION = ALL Nos. are Prime

N = 50 ⟶ [1 to 50]



PSEUDO CODE

```
getallPrime(N){
    bool P[N+1] = {T};
    P[0] = P[1] = F;
    for(i=2; i<=n; i++){
        if(P[i]==T){
            for(j=2*i; j<=n; j+=i){
                P[j] = F;
            }
        }
    }
}
```

↳ OPTIMIZATION (Sieve of Eratosthenes)

Q what we have done on these nos?

2 → 4, 6, 8, 10, 12, 14 .....

3 → 3, 6, 9, 12, 15, ...

5 → 10, 15, 20, 25, 30, ...

7 → (7x2), (7x3), (7x4), (7x5), (7x6), (7x7)

Already Marked ..

(7x8), ...

Observation

↳ Can We observee that before (7x7) other no is multiplication with 7 would have already marked that No.

Eg:- 11 → (11x2) (11x3) ..... (11x10) (11x11) ....

Already Marked

PSEUDO CODE

```
get allPrime(N) {
    bool P[N+1] = {T};
    P[0] = P[1] = F;
    for( i=2; i <= √n; i++){
        if( P[i] == T){
            for( j= i*i; j<=n; j+=i){
                P[j] = F;
            }
        }
    }
}
```

TIME COMPLEXITY [Slight cheating]

outer loop [i]		Inner loop [j]
	(Rough)	
2	$\approx$	N/2
3	$\approx$	N/3
5	$\approx$	N/5
⋮		

So, $TC = \left[\frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \dots \right]$

$= N \left[\underbrace{\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \dots}_{\text{Harmonic progression of primes.}} \right]$

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

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

$SC = O(N)$

Beautiful Observation

Let $N = 2^{64} \approx 10^{18}$

Previous Idea	Current Idea
$TC \rightarrow O(N\sqrt{N})$	$TC \rightarrow O(N \log\log N)$
$= 10^{18} \sqrt{10^{18}}$	$= 10^{18} (\log \log 2^{64})$
$= 10^{18} * 10^{9} = 10^{27}$	$= 10^{18} (\log 64)$

$\leadsto 10^{18} * 6$

$\leadsto 6 \times 10^{18}$

Observation :- Current Idea is 10 Gs time better than previous Idea

if $N = 10^{18}$

Break → 10 : 10 → 10 : 20

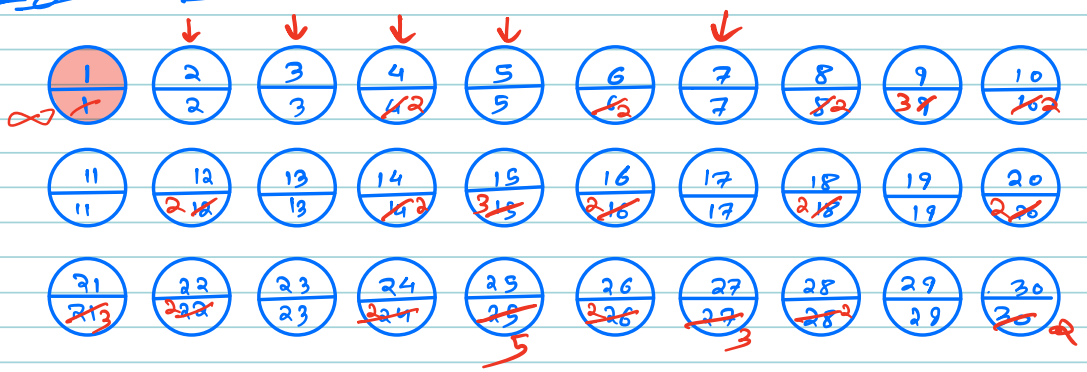
question :- Smallest Prime factors (SPF)

Given N, return the smallest prime factors of all numbers from 2 to N.

Ex:- N = 10 → 2, 3, 4, 5, 6, 7, 8, 9, 10

Ans SPF → 2 3 2 5 2 7 2 3 2

Eg 2 N = 30



Observation

=> For prime no :- SPF will be no. itself.

=> we will not update while updating SPF if its already updated by someone else.

PSEUDO CODE

```
int[] SPFCreation ( N ) {
    int[] SPF (N+1); // initially SPF[i]=i;
    SPF[0] = SPF[1] = ∞;
    for ( i=2; i <= √N ; i++) {
        if ( SPF[i] == i ) {
```

```
for(j=i*i; j<=n; j+=i){
    if(SPF[j]==j){
        SPF[j]=i;
    }
}
}
return SPF;
}
```

TC → O(N log log N)
SC → O(N)

Prime Factorization

↳ Representing a no. in multiple of power of unique prime number.

Eg:- N = 48

2	48
2	24
2	12
2	6
3	3
	1

$\Rightarrow N = 48 \rightarrow 2^4 * 3^1$

$\Rightarrow N = 45 \longrightarrow 3^2 * 5^1$

$\Rightarrow N = 300 \longrightarrow 2^2 * 3^1 * 5^2$

⇒ CONCEPT / OBSERVATION

if you have a no. n, whose prime factorization is :-

$$p_1^{a_1} * p_2^{a_2} * p_3^{a_3} * \ldots\ldots * p_y^{a_y}$$

No. of factors :- $(a_1+1) * (a_2+1) * (a_3+1) * \ldots * (a_y+1)$

Eg:- $N = 48 \rightarrow 2^4 * 3^1 \rightarrow (4+1)*(1+1) = 5*2 = 10$

$\Rightarrow N = 45 \rightarrow 3^2 * 5^1 \rightarrow (2+1)(1+1) = 3*2 = 6$

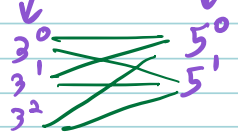
$\Rightarrow N = 300 \rightarrow 2^2 * 3^1 * 5^2 \rightarrow (2+1)*(1+1)*(2+1)$
$= 3*2*3 = 18$

Explanation/Reasoning

$n = 45 \rightarrow 3^2 * 5^1 = (2+1)(3+1) = 6$

Factors

1, 3, 5, 9, 15, 45



Question :- Count of No. of Factors

Given N, No. of Factors of each no. from 1 to N.

Ex N = 10 →

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

Ans

Brute force

↳ For all nos. from 1 to n, find count of factors by $\sqrt{N}$ method.

TC → $O(N\sqrt{N})$

SC → $O(1)$

↳ OPTIMIZATION

Ex 1 $n = \frac{49}{7}^{7} = \frac{7}{7} = 1$ (SPF → SPF)

$7 \xrightarrow{\text{Power of 7}} \cancel{1}\ 2$

Ex 2 $n = \frac{48}{2} = \frac{24}{2}^{12} = \frac{12}{2}^{6} = \frac{6}{2}^{3} = \frac{3}{3} = 1$

$2 \xrightarrow{\text{Power of 2}} \cancel{1}\ \cancel{2}\ \cancel{3}\ 4$

$3 \xrightarrow{\text{Power of 3}} 1$

Ex 3 $n = 54 \Rightarrow \frac{54}{2}^{27} = \frac{27}{3}^{9} = \frac{9}{3}^{3} = \frac{3}{3} = 1$

$2 \rightarrow 1$

$3 \rightarrow \cancel{1}\ \cancel{2}\ 3$

PSEUDO CODE

```
SPF[N+1]    // TODO
for ( x = 1 to N ) {
    X // --> Find the Count of Factors
    hm <int, int>;
    while ( x > 1 ) {
        int D = SPF[x];
        if ( D is in hm ) {
            hm[D] += 1;
        } else {
            hm[D] = 1;
        }
        X = X/D;
    }
}
```

$(N \log \log N)$

$\log_2 N$ (worst case when D is 2)

$\log_2 N$

DRY RUN

~~x = 48 24 12 6 3~~ 1

D = ~~2 2 2 3~~

Map: { 2 → ~~1 2 3~~ 4, 3 → 1 }

ans = 1

```
for (x : hm.keySet()) {
    ans = ans * (hm.get(x) + 1);
}
Print (ans); <-- Count of Factors
```

TC → $O(N \log \log N + N(2 \log_2 N))$

$\Rightarrow O(N \log_2 N)$

SC → $O(N) + O(\log_2 N)$

$O(N)$