DSA

- → Arrays
- → Maths
- → Bit Manipulations
- → Recursion
- → Sorting
- → Binary Search
- → Two pointers
- → Hashing
- → Pattern Matching
- → String Problems
- → Linked List
- → Stacks / Queue / DeQue
- → BT / BST
- → Tries
- → Segment Trees
- → Heaps
- → Greedy Algorithms
- → Back Tracking
- → Dynamic Programming
- → Graphs.

Prime Numbers: Number which has 2 factors, 1 & itself

$O(N)$ — $O(\text{sqrt}(N))$

$N = 12$

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

$i <= N/i$

$i <= \sqrt{N}$

$N = 36$

i	N/i
1	36
2	18
3	12
4	9
6	6
9	4
12	3
18	2
36	1

```
// bool isPrime(N){
    cnt = 0, if(N==1) return false
    i = 2; i <= √N; i++){
        if(N % i == 0){
            return false
        }
    }
    return true;
}
```

If a number a prime it won't have a single factor from [2 √N] }

→ TC: $O(\sqrt{N})$

Given N Find all the primes from 1 - N

N = 10 : [1 10] : 2 3 5 7

Sol: For every number from [1 N], check prime or not with isprime( )

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

```
cnt = 0;
i = 1; i <= N; i++) {
    if( isprime(i)) {
        cnt++
    }
}
```

// N = 50

~~0~~ ~~1~~ 2 3 ~~4~~ 5 ~~6~~ 7
~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~
~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~ ~~21~~
~~22~~ 23 ~~24~~ ~~25~~ ~~26~~ ~~27~~ ~~28~~
29 ~~30~~ 31 ~~32~~ ~~33~~ ~~34~~ ~~35~~
~~36~~ 37 ~~38~~ ~~39~~ ~~40~~ 41 ~~42~~
43 ~~44~~ ~~45~~ ~~46~~ 47 ~~48~~ 49
~~50~~

bool p[51] = T

	number we
2 →	4
3 →	9
5 →	25
7 →	49

.

}

// Given N

$i * i <= N$

bool p[N+1] = T

} N=100

p[0] = p[1] = F

i = 2 ; i <= N ; i++) {

instead

$i <= \sqrt{N}$

$i = 2 : 4$

$i = 3 : 9$

$i = 5 : 25$

$i = 7 : 49$

$i = 11 : 121$

$i = 13 : 169$

```
if( p[i] == True) {
    // i is a prime.
    // we need iterate in
       multiples of i.
    j = 2*i; j <= N; j = j+i) {
        p[j] = False
    }
}
}
```

instead we can also use i*i

⟹ Seive of Erosthenes

$$TC \Rightarrow N/2 + N/3 + N/5 + N/7 + N/11$$

$$\Rightarrow N[\underbrace{1/2 + 1/3 + 1/5 + 1/7 + 1/11 + \cdots 1/p}_{\text{Sum of reciprocals of primes} = \log(\log N)}]$$

↳ last prime <= N

$\log_2 N$ $\quad N = 2^{30}$

$\log_2 \log_2 N = \log_2 30$

$= 5$

$$TC \Rightarrow N[\log(\log N)] \approx O(N) \qquad SC: O(N)$$

20) Given N, Find smallest prime factor for all numbers 2-N

Ex: 10 → spf: 2

15 → spf: 3

17 → spf: 17.

N = 10:

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

N = 25 : spf[] →

4 != spf[4] 4 is not prime

	~~1~~	2	3	4	5	6	7	8	9	10	11	12	13	14
spf		2	3	~~4~~ 2	5	~~6~~ 2	7	~~8~~ 2	~~9~~ 3	~~10~~ 2	11	~~12~~ 2	13	~~14~~ 2

	15	16	17	18	19	20	21	22	23	24	25
spf	~~15~~ 3	~~16~~ 2	17	~~18~~ 2	19	~~20~~ 2	~~21~~ 3	~~22~~ 2	23	~~24~~ 2	~~25~~ 5

```
// spf[N+1] = {// initialize spf[i] = i }

i = 2 ; i <= √N ; i++) {

        if( spf[i] == i) {        → i is prime.

            j = i*i ; j <= N ; j = j+i

                if( spf[j] == j) {        or   spf[j] = min(i, spf[j])
                    spf[j] = i;
                }
            }
        }
}
```

3Q) Count no: of divisors

product of power of primes → prime factorization

Ex1 $n = 72 = 2^3 * 3^2$

$2^0 * 3^0 = 1$	$2^0 * 3^1 = 3$	$2^0 * 3^2 = 9$	$[2^0\ 2^1\ 2^2\ 2^3]$
$2^1 * 3^0 = 2$	$2^1 * 3^1 = 6$	$2^1 * 3^2 = 18$	$[3^0\ 3^1\ 3^2]$
$2^2 * 3^0 = 4$	$2^2 * 3^1 = 12$	$2^2\ 3^2 = 36$	
$2^3 * 3^0 = 8$	$2^3 * 3^1 = 24$	$2^3\ 3^2 = 72$	$4 * 3 = 12$

Ex2: $n = 600 = 2^3 * 3^1 * 5^2 \Rightarrow \{(3+1) * (1+1) * (2+1)\}$

$\Rightarrow \{24\}$

// generalization → For crap!

$P_1, P_2 .. P_y$ are primes

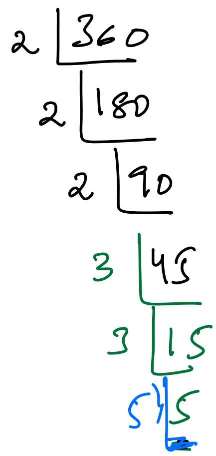
$$N = P_1^{n_1} * P_2^{n_2} * P_3^{n_3} \cdots P_y^{n_y}$$

Total factors $= (n_1+1)(n_2+1)(n_3+1) \cdots (n_y+1)$

10:50 break

// Break

N = 360 → $2^3 * 3^2 * 5^1$



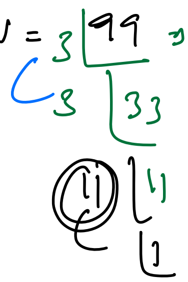
$\Rightarrow (3+1)*(2+1)*(1+1)$

$\Rightarrow (4)(3)(2) \Rightarrow 24$

N = 99 ⇒ 3 ⇒ (2+1)(1+1)

⇒ (6)

11 ⇒ 11



// get Total factors for N

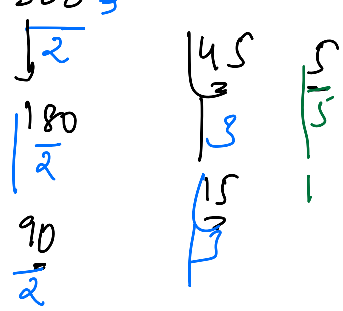
Total = 1 → TC: log N

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

For N get it's prime fact

```
list<int> fact
while (N > 1) {
    x = spf[N]
    fact.add(x)
    N = N/x
}
```

N = 360 ⇒ fact {2, 2, 2, 3, 3, 5}



3Q) Given N for every number from 1-N, get number of factors.

N : 10 → {1 2 3 4 5 6 7 8 9 10}

{1 2 2 3 2 4 2 3 3 4}

1) Step 1: Create spf[N+1], write code to create spf[N+1]

```
cnt [N+1] ;
cnt [1] = 1 ;
i = 2 ; i <= N ; i++ ) {
    n = i ; Total = 1
    while( n > 1) {
        p = spf[n]
        c = 0
        while(n % p == 0){
            c = c+1;
            n = n/p
        }
        Total = Total * (c+1)
    }
    cnt [i] = Total
}
```

overall TC:

TC: N * log N +

TC to get spf[]

SC: O(N) → spf[]

4(8) Are prime finite or infinite ?

Assume they are N primes : $P_1, P_2, P_3, P_4, P_5 . P_N$

consecutive numbers

// Multiplication of all N primes

$$\left\{\begin{array}{l} x = P_1 * P_2 * P_3 * P_4 \ldots P_N \\ y = x + 1 \end{array}\right.$$

$\gcd(x, y) = 1$

↳ No common divisor between x & y

prime factors x:

P_1
P_2
P_3
$\vdots$
P_N

→ None of these primes can divide y.

That means y have to be a prime.

↳ Contradiction

// Doubts :

2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1}

2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 21

$N + \log_2(Max) \} =$

1

→ gcd of N array Element → $O(N + \log_2^{Max})$

Max

g = Max

i=0; i<N; i++) {

g = gcd(g, arr[i])

}

//2

// Ex:

gcd(a, b) : $\log_2(\max(a, b))$

{until b reaches = 1}

Total outer   Total inner

$N \cdot \log_N(\max(arr))$

```
g = ar[0]
i = 0; i < N; i++){
    g = gcd(g, ar[i])
}
```

1

g = $\boxed{gcd(g, ar[i+1])}$

1

g = gcd(g, ar[i+2])

t

// Segmented Sieve. {optional}

Given a & b calulate no: of primes in range [a b]

$1 <= a <= b <= 10^{10}$

$b - a <= 10^5$

1) Get all primes from $\underline{10}^5$ store in list

2) Now ch[b-a] = T;

i = 0; i < l.size(); i++) {

α =

```
    int p = l[i]
    int mul;
    if(a % mul == 0){ mul = a }
    else { mul = a%p*p + p }
    if( mul == p} { mul = 2p}
    j = mul; j <= b; j = j+p) {
        ch[j-a] = F
    }
}
```

// Segmented Sieve

Given a & b, we need to get all prime from a to b

$\boxed{a = 100 \quad b = 130}$ → Exactly how many : 31

⇒ Sieve of Erathanes : p[131] } → memory wastage

// count no: of True are there index [100 130]

// $\sqrt{130}$ = 11 Till 11 = 2 3 5 7 11

100 102

p[31] :

~~100 0~~	101 1	~~102 2~~	103 3	~~104 4~~	~~105 5~~	~~106 6~~	107 7
~~108 8~~	109 9	~~110 10~~	~~111 11~~	~~112 12~~	113 13	~~114 14~~	~~115 15~~
~~116 16~~	~~117 17~~	~~118 18~~	~~119 19~~	~~120 20~~	~~121 21~~	~~122 22~~	~~123 23~~
~~124 24~~	~~125 25~~	~~126 26~~	127 27	~~128 28~~	~~129 29~~	~~130 30~~	

$\Rightarrow$ $100 \% 2 = 0$ ✓ 100 is first multiple of 2

$100 \% 3 \neq 0$ $\quad \underbrace{\left(\frac{100}{3}\right)}_{0} * 3 = 99 + 3 = 102$

$100 \% 5 == 0$ ✓ 100 is first multiple of 5

$100 \% 7 \neq 0$ $\quad \underbrace{\left(100/7\right)}_{0} * 7 + 7 = 105$

$100 \% 11 \neq 0$ $\quad \underbrace{\left(100/11\right)}_{99} * 11 + 11 \Rightarrow \boxed{110}$

// Given a & b find all prime from [a, b]

Constraints

{ $1 <= a <= b <= 10^{10}$ }

$[b - a] <= 10^5$

→ Use Sieve works

// We need all prime till $\sqrt{10^{10}} = 10^5$ → { Linklist l } → $\approx 10^5$ iterations

```
bool ch[b-a+1] = T;

i=0; i< l.size(); i++){  → {n}
    p = l[i];          p is a prime number.
    // we need to get first multiple mul;
    if( a%p == 0) { mul = a }
    else { mul = (a/p)*p + p)
    if( mul == p) { mul = 2p }  → {Edge case}
    j = mul ; j <= b; j = j+p) {
        ch[ j-a] = false;
    }
}
```

// Edge case : $\begin{array}{cc} a & b \\ \end{array}$ $[1 \quad 50]$

list of prime : v : 2 3 5 7

what's the first multiple of 2.

$a \% 2 == 0$ } $(a/2)*2+2$

$= (1/2)*2$

$= 0+2 = 2$

$(2/7)*7+7$

$= 0+7$

$= 7$

$a \% 3 \neq 0$ } $(a/3)*3+3$

$= 0+3$

$= 3$

$\frac{b-a}{p_1} + \frac{b-a}{p_2} + \ldots \frac{b-a}{p_n}$

$N = 10^7$

$\log \log 10^7$

≈ 10

$(10)/r$

$b-a\left[\frac{1}{p_1} + \frac{1}{p_2} + \ldots \frac{1}{p_n}\right]$

$= \log \log n$

$$\Rightarrow \quad \frac{b-a}{p_1} + \frac{b-a}{p_2} + \cdots \frac{b-a}{p_{10^4}}$$

$$b-a \left[\underbrace{\frac{1}{p_1} + \frac{1}{p_2} + \cdots \frac{1}{p_{10^4}}}_{} \right]$$

$$b-a \left\{ \log_2 (\log_2 (10^4)) \right\}$$

$$b-a \} \approx 10 \}$$

$$b$$

$$10^6 - a \quad 10^6$$