Prime Numbers

Today's content

→ Prime number Intro

→ Get all primes from 1 to N

→ Print smallest prime factor for 2 to N

→ Prime factorisation

→ Get the no. of factors/divisors

Prime numbers

→ No. having only two factors
└ 1 & itself

Eg :- 2, 3, 5, 7, 11

Q Given a no., we need to check if it is prime or not.

Ans 1 = Count the no. of factors

factors == 2 ⟶ prime

factors > 2 ⟶ not prime

```
boolean checkprime ( int n)
    count = 0

    for( i=1 ; i ≤ √n ; i++){
        if (n%i ==0){
            if (i==n/i) count ++;

            else count += 2
        }
    }
    if (count == 2) print (prime);

    else print (not prime )
}
```

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

SC : $O(1)$

Q Given N, print all the prime no. from 1 to N

N = 10 ⇒ 2, 3, 5, 7

N = 20 ⇒ 2, 3, 5, 7, 11, 13, 17, 19

BF ⇒ Iterate from 1 to N & check if a no. is prime or not.

```
void printallprime(int n)
{
    for (i=2; i≤n; i++){
        if (checkprime(i) == true){
            print(i);
        }
    }
}
```

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

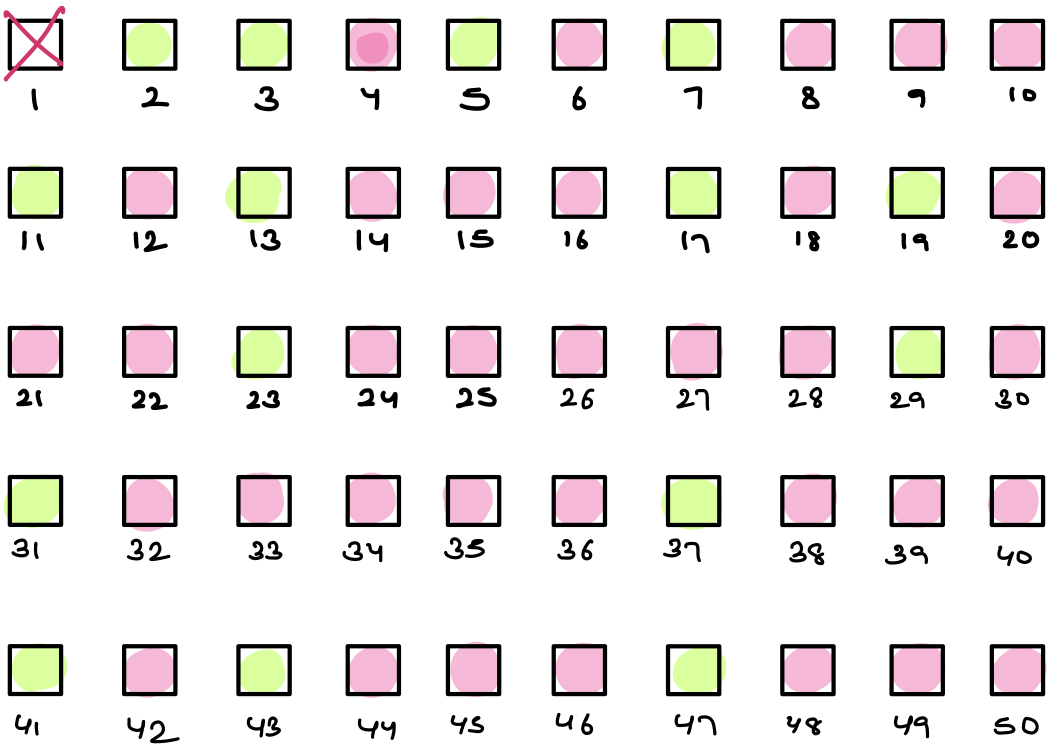
SC: $O(1)$

* Idea 2 Sieve of eratosthenes

↳

Find all prime no. from 1 to 50

Ass → Every no. is a prime no.



```
void printallprimes (int n)
{
    boolean [] prime = new boolean[n+1];

    //Mark every idx as true

    Arrays.fill(prime, true);

    prime[0] = false
    prime[1] = false;

    for (i=2; i*i<=n; i++)
    {
        // iterate on multiples of i if
            i is prime

        if (prime[i] == true){
            for (j=i*i ; j<=n ; j=j+i ){
                prime[j] = false;
            }
        }
    }

    for (i=2; i<=n; i++){
        if (prime[i]==true) print(i);
    }
}
```

i	mul of i, which we are going to mark fal
2	$2*2$ $2*3$ $2*4$ $2*5$ $2*6$ ...
3	$3*2$ $3*3$ $3*4$ $3*5$ ...
4	______
5	$5*2$ $5*3$ $5*4$ $5*5$ $5*6$ ...

* for every prime → start marking false from $i*i$

i [$2 \to n$]	$j = [i*i \to n]$
2	$\simeq n/2$ iterations
3	$\simeq \frac{n}{3}$ iterations
4	X
5	$\simeq \frac{n}{5}$ iterations
$\sqrt{n}$	$j = [(\sqrt{n})^2 \to n]$ 1 iteration
$(\sqrt{n}+1)$	$j = ((\sqrt{n}+1)^2 \to n]$ $(n+2\sqrt{n}+1 \to n)$ 0 iteration
$(\sqrt{n}+2)$	0 iterations

$\vdots$

$\vdots$

O iterations

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

: $N\underbrace{\left(\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \cdots\cdots\right)}_{\text{sum of all reciprocals of prime no.}}$

$= N\left(\log(\log N)\right)$

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

SC : $O(n)$

Q3 Given N. Return the smallest prime factor for all no. from 2 to n

N = 10	2	3	4	5	6	7	8	9	10
Ans	2	3	2	5	2	7	2	3	2

* spf for ele 1 → 50

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

* To identify prime if (ar[i] == i)

Obs → Do not update the preupdated valu

```
int [] spf = new int [n+1]
```

01. all ar[i] = i

```
for (i=0; i<=n; i++){
    spf[i] = i;
}

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(1)

Question asks to return spf

10:10 → 10:20 pm

* Prime factorisation

n = 48

2	48
2	24
2	12
2	6
3	3
	1

$n = 48 \Rightarrow 2 * 2 * 2 * 2 * 3$

$= 2^4 * 3^1$

No. of divisors $= (4+1) * (1+1)$

$= 5 * 2 = 10$

n=48 - 1, 2, 3, 4, 6, 8, 12, 16, 24, 48

n = 300

2	300
2	150
3	75
5	25
5	5
	1

$n = 300 = 2 * 2 * 3 * 5 * 5$

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

No. of divisors $= (2+1) * (1+1) * (2+1)$

$= 3 * 2 * 3$

$= 18$

Q Given a no. n, assume its prime factorisation as

$$n = i^{a_1} * j^{a_2} * k^{a_3} * l^{a_4} \dots z^{a_x}$$

Calculate no. of divisors

No. of divisor/factors $= (a_1+1) * (a_2+1) * (a_3+1) * \dots * (a_x+1)$

5	25
5	5
	1

$25 = 5^2$

No. of divisors $= (2+1) = 3$

$25 \Rightarrow 1, 5, 25$

* Given a number N. For all the numbers from 1 to n, get the no. of factors/divisors

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

Ans = { 1 2 2 3 2 4 2 4 3 4 }

Idea → For all no. → consider the prime factorisation

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

x

$360 = 2^3 * 3^{2} * 5^1$

Map

K	V
2	→ 3
3	→ 2
5	→ 1

No. of divisors $= (3+1) * (2+1) * (1+1)$

$= 24$

* pseudocode

01. Create & fill the spf ar[n+1] → $O(n \log(\log n))$

$+$

$n * \log n$

```
for (i=1 ; i<=n; i++)
    HM<I, I> map = fill (i, spf )
    int ans = 1
    for ( int key : map. keySet()){
        int v= map.get(key)+1
        ans = ans * v;
    }
    print (ans);
}
```

```
public HM<I,I> fill(int x, int[] spf)
{
    HM<I,I> hm = new HM<>();

    while (x > 1) {

        if (hm.containskey(spf[x]);
        {
            hm.put(spf[x], hm.get(spf[x])+1);
        }
        else {
            hm.put(spf[x], 1);
        }
        x = x/spf[x]
    }
}
```

$\underbrace{\texttt{hm.get(spf[x])}}_{\text{old freq}}$

$\frac{x}{2}$ $\frac{x}{3}$ $\frac{x}{5}$ $\frac{x}{7}$