PRIME

Content

- Count of Divisors
- Check if the given N isPrime
- Find all prime numbers from [1 - N]
- Find the count of divisors of all numbers [1 - N]
- Find smallest prime factor (spf) of all [1 - N]
- Find the count of divisors of all numbers using spf.

Count of Divisors

Given N find count of all divisors.

10 ⟶ 1 2 5 10 ⟶ (4)

- Bruteforce — Go to all numbers from 1 to N and check if its a factor.

32 ⟶ 1, 2, 4, 8, 16, 32

Optimal

TC : O(sqrt(N))

```
int countFactors (N) {
    count = 0
    for (i = 1; i*i <= N; i++) {
        if (N % i != 0) {
            continue
        }
        first = i
        second = N/i
        if (first != second) {
            count += 2
        }
        else {
            count += 1
        }
    }
    return count
}
```

Is Prime — Exactly 2 factors

N = 10 not prime {1 2 5 10} more than 2

N = 7 prime {1 7}

```
bool isPrime(N) {
    return countFactors(N) == 2
}
```

- Print all primes from 1 - N

N = 10 ⟶ Output {2, 3, 5, 7}

Idea 1 Brute force

loop from 1 to N

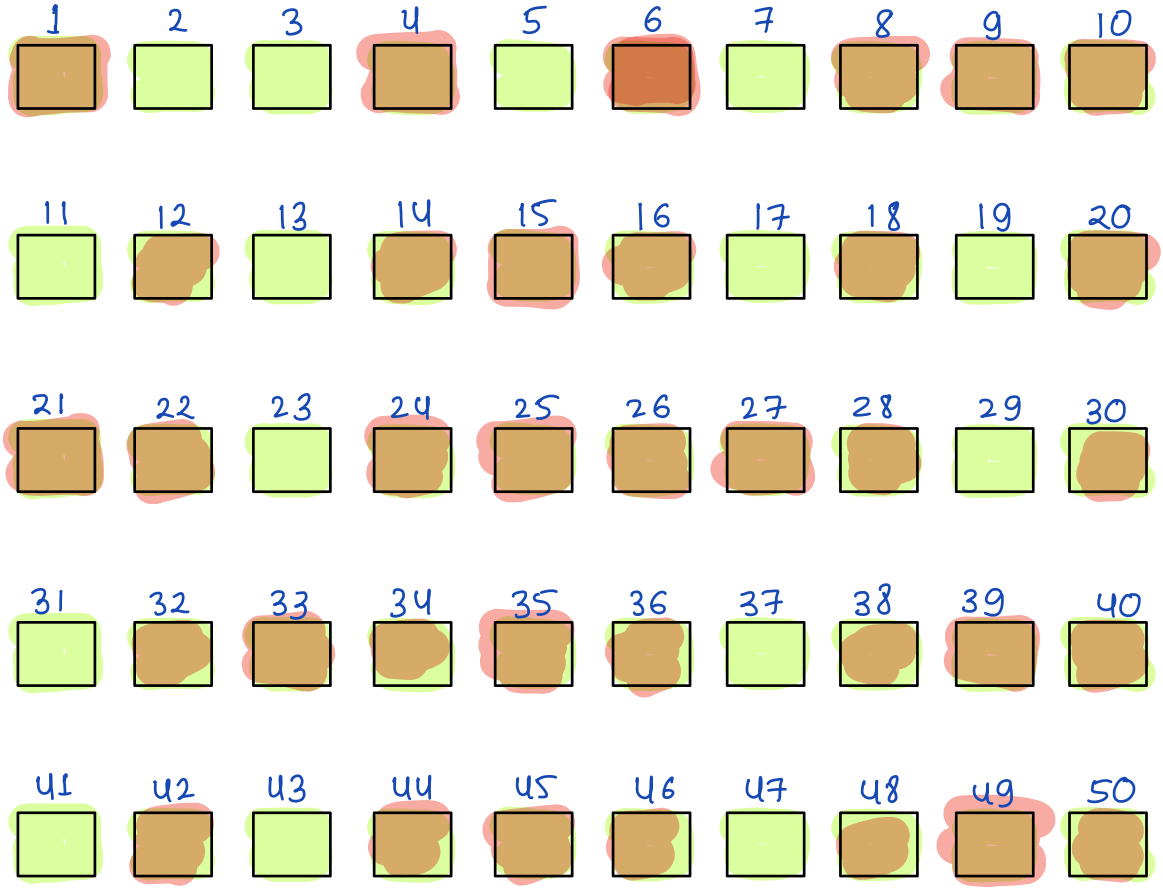
call isPrime(i) ⟶ print

⟶ dont print

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

Idea 2

Say we need all primes from 1-50

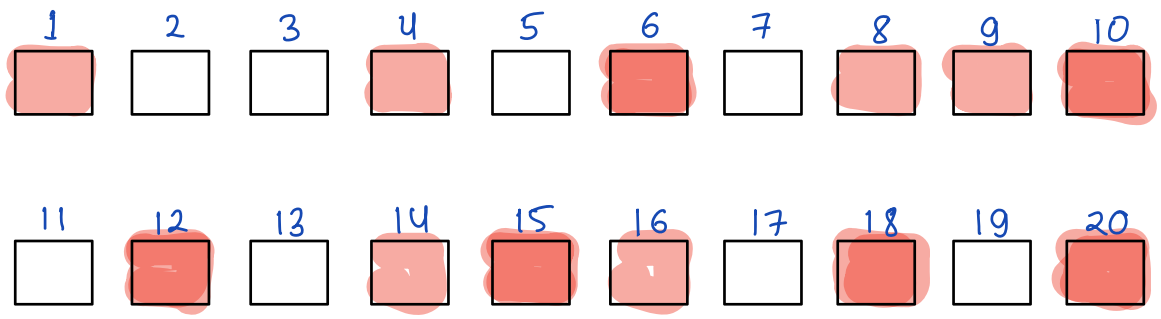


Pseudo code

Print all primes in range [1, N]

Sieve of Eratosthenes

```
allPrime ( int N ) {
    bool sieve [N+1]      // Init with all true
    sieve [0] = false
    sieve [1] = false
    for (i = 2 ; i <= N ; i++) {
        if(sieve [i]) {
            for ( j = ix2 ; j <= N ; j += i ) {
                sieve [j] = false
            }
        }
    }

    for ( i -> 2 to N) {
        if (sieve [i]) {
            print (i)
        }
    }
```



Time Complexity

i	j	iterations
2	multiples of 2	N/2
3	multiples of 3	N/3
4	X	
5	multiples of 5	N/5
6	X	
7	multiples of 7	N/7
8	X	
9	X	

TC : $\frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \frac{N}{7} + \frac{N}{11} \ldots\ldots$

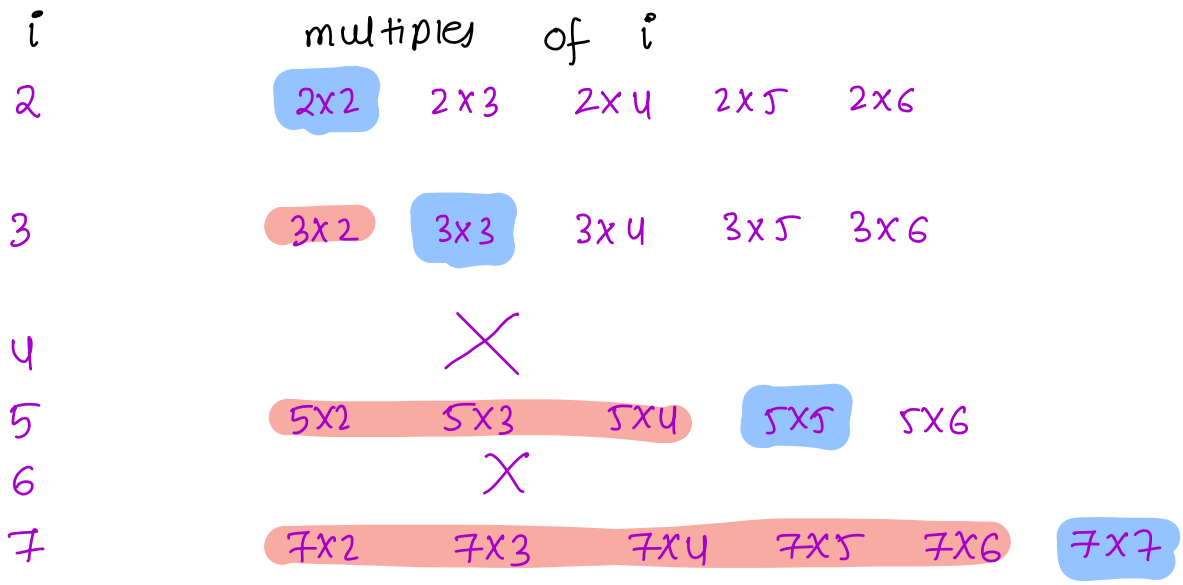
$N \left\{ \frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} \ldots\ldots \frac{1}{N} \right\}$

$O(N \{ \log(\log N) \})$

sum of reciprocals of all primes = $\log(\log N)$

Optimization to the above idea

N = 36




```
allPrime ( int N ) {
    bool sieve [N+1]      // Init with all true
    sieve [0] = false
    sieve [1] = false
    for (i = 2 ; i*i <= N ; i++) {
        if(sieve [i]) {
            for ( j = i*i ; j <= N ; j += i ) {
                sieve [j] = false
            }
        }
    }
```

i	j	inner loop will execute
$\sqrt{N}$	N	yes
$\sqrt{N}+1$	$(\sqrt{N}+1)^2$ $N+2\sqrt{N}+1$	

TC : TODO

Find no. of factors for all 1 to N

N = 10 :	1	2	3	4	5	6	7	8	9	10
# factors	1	2	2	3	2	4	2	4	3	4

Brute force : $i \longrightarrow 1$ to N

print (countFactors (i))

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

Idea 2 :

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

```
int[] allFactors(int n) {
    cf[N+1]   // init with 1 since 1 if factor of all.

    for(i=2; i<=N; i++) {
        for(j=i; j<=N; j+=i) {
            cf[j] += 1
        }
    }

    return cf
}
```

Time Complexity

i	j	iterations
2	multiples of 2	N/2
3	multiples of 3	N/3
4	multiples of 4	N/4
5	multiples of 5	N/5
6	multiples of 6	N/6
7	multiples of 7	N/7
8	multiples of 8	N/8
9	multiples of 9	

TC : $\frac{N}{2} + \frac{N}{3} + \frac{N}{4} \ldots\ldots 1$

$N\left\{\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} \ldots\ldots \frac{1}{N}\right\}$

TC : N log N

Sum of reciprocals of all natural numbers.

Break : 10 : 14 pm

SPF

Find smallest prime factor for all numbers from 1 - N

N = 10 :	1	2	3	4	5	6	7	8	9	10
spf[] :	0	2	3	2	5	2	7	2	3	2

0 → no prime factor

Idea N = 50

1	2	3	4	5	6	7	8	9	10
0	2	3	2	5	2	7	2	3	2

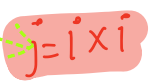
11	12	13	14	15	16	17	18	19	20
11	2	13	2	3	2	17	2	19	2

21	22	23	24	25	26	27	28	29	30
3	2	23	2	5	2	3	2	29	2

31	32	33	34	35	36	37	38	39	40
31	2	3	2	5	2	37	2	3	2

41	42	43	44	45	46	47	48	49	50
41	2	43	2	3	2	47	2	7	2

```
int[] allSpf(int N) {

    spf[N+1]      // init
    for (i = 0; i <= N; i++) {
        spf[i] = i
    }

    spf[1] = 0
    spf[0] = 0

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

                spf[j] = min(spf[j], i)

            }
        }
    }

    return spf
}
```



Time Complexity

i	j	iterations
2	multiples of 2	N/2
3	multiples of 3	N/3
4	X	
5	multiples of 5	N/5
6	X	
7	multiples of 7	N/7
8	X	
9	X	

TC : $\frac{N}{2} + \frac{N}{3} + \frac{N}{5} + \frac{N}{7} + \frac{N}{11} \ldots\ldots$

$N \left\{ \frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} \ldots\ldots \frac{1}{N} \right\}$

$O(N \{ \log(\log N) \})$

sum of reciprocals of all primes = $\log(\log N)$

Prime Factorization

2	12
2	6
3	3
	1

$12 = 2^2 \times 3^1$

2^2: 0 to 2 ; 3^1: 0 to 1

Factors 12

1, 2, 3, 4, 6, 12

at max 4 can we 2 twice and 3 once.

6

$2^0 \times 3^0 = 1$

$2^1 \times 3^0 = 2$

$2^0 \times 3^1 = 3$

$2^2 \times 3^0 = 4$

$2^1 \times 3^1 = 6$

$2^2 \times 3^1 = 12$

$$N = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k}$$

$$= (a_1+1) \times (a_2+1) \cdots (a_k+1)$$

$32 = 2^5$

$= 6$ # of factors.

1 2 4 8 16 32

$45 = 3^2 \times 5^1$

$= 6$ # of factors.

1 3 5 9 15 45

2	32
2	16
2	8
2	4
2	2
	1

3	45
3	15
5	5
	-

Find no. of factors for all 1 to N using SPF

spf [52] = 2
spf [26] = 2
spf [13] = 13

52
26
13
1

spf [52] = 2

⟹ take freq of prime factorization.

2 : 2

13 : 1

factors = (2+1) x (1+1)

Assume spf [N+1] is already present.

```
int CFUSPF ( N ) {
    factors = 1
    freq [N+1]
    while ( N > 1 ) {                 // Log N
        sp = spf [N]
        freq [sp] ++
        N = N / sp
    }
    for ( i = 2 → N ) {               // O(N)
        factors *= (freq [i] + 1)
    }
```

TC : O(N)

see optimized code below.

N → N/2 1

```
    return factors.
}

void allFactorsSPF(N){
    for(i -> 1 to N){ ---> O(N)
        print(CFUSPF(i)) ---> O(log N)
    }
}
```

TC : // Create SPF[] $O(N \log N(\log N))$ + // allfactor SPF $O(N \log N)$

Optimised

```
int CFUSPF(N){                                   O(log N)
    factors = 1

    while(N>1){
        sp = spf[N]
        freq = 0

        while(N%sp==0){
            freq++
            N = N/sp
```

2	32
2	16
2	8
2	4
2	2
	1

```
        }
        factor = factor x (freq+1)
    }

    return factors.
}
```