01 Print divisors of a number

N = 36

[1, 2, 3, 4, 6, 9, 12, 18, 36]

$1 \times 36 = 36$
$2 \times 18 = 36$
$3 \times 12 = 36$
$4 \times 9 = 36$
$6 \times 6 = 36$

Both all divisors.

TC = O(sqrt(n))
SC = O(N).

02 Check if a number is prime or not

No is divisible by itself and by 1 and no. of divisors = 2

```
cnt = 0
for (i = 1 -> sqrt(N)) {
    if (N % i == 0) {
        cnt = cnt + 1
        if (N / i != i)
            cnt = cnt + 1
    }
}
```

TC = O(sqrt(n))
SC = O(1).

03 Prime factors of given N.

Eg. N = 60

1, (2), (3), 4, (5), 6, 10

12, 15, 20, 30, 60

Eg. N = 30

5, 7

Eg. N = 780

2, 3

04 Power exponentiation

- Brute force

run loop and multiply again and again.

```
Pow (x, n) {
    ans = 1
    while (n > 0) {
        if (n % 2 == 1) {
            ans = ans * x;
            n = n - 1;
        }
        else {
            n = n / 2;
            x = x * x;
        }
    }
    return ans;
}
```

Now $5^{-2} = 1/5^2$

→ Simply compute they and return $1/5^2$.

- If we have to handle double wala case. Then put double.

$TC = O(\log_2 n)$

$SC = O(1)$.

$x = 2$

$n = 5$

Now, Pow (2, 5)

$= 32$.

Now $x = 2 \to 4$ and $n = 20 \to 10$

$2^{21} \to (2^{20}) \cdot 2 \to (2^2)^{20/2}$

ans = 1 × 2 $= (4)^{10}$

reduction possible.

Now, $4^{10} \to (4^2)^{10/2}$

$= (16)^5$

Now, $16^5 \to 16 \times 16^4$

ans = 1 × 2 × 16

$16^4 = ((16)^2)^{4/2}$

$= (256)^2$

$(256)^2 = ((256)^2)^{2/2}$

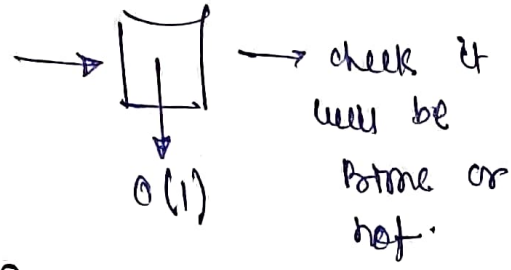
$= (65536)^1$

So, ans = 1 × 2 × 16 × 65536.

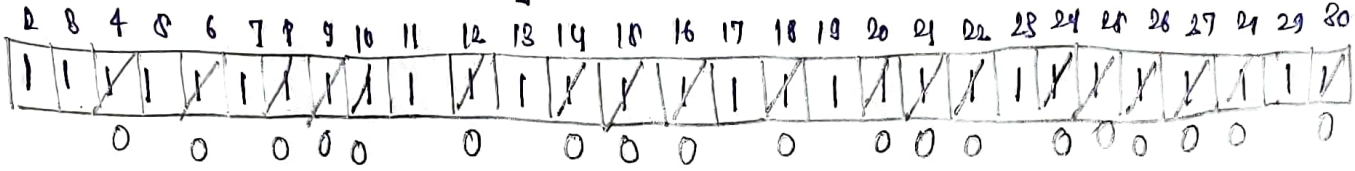
- When Power is odd, remove and add 4 to simply in ans.

05 Sieve of Eratosthenes

Given a no. N, print all primes from till N.



Q. N = 30 Prime [31]



Here all are marked with 0 and 1, 0 shows not prime one (1) wala shows these are primes.

```
Prime(N) {
    Prime [N+1] → (2 → N) as 1
    For (i = 2 → N) Prime [i] = 1
    For (i = 2 → N) {                    // N → √N.
        if (Prime [i] == 1) {
            [ For (j = 2*i; j <= N; j += i) {     // i*i
                Prime [j] = 0;
            }
        }
    }
    For (i = 2 → N) {
        if (Prime [i] == 1) {
            Print (i);
```

Now optimised

- tc = O(N log(log N)) + O(N) + O(N)
- sc = O(N).

2×2	4×2	5×2	7×2
2×3	6×3	5×3	7×3
2×4	8×4	5×4	7×4
2×5	3×5	5×5	7×5
⋮	⋮	⋮	⋮

MATH FOR DSA

01.) PRINT DIVISOR

```
class Solution {
  public:
    void print_divisors(int n) {
        vector<int>divisors;
        for(int i=1 ;i*i<=n;i++)
        {
            if(n%i==0)
            {
                divisors.push_back(i);
                if(n/i!=i)
                {
                    divisors.push_back(n/i);
                }
            }
        }
        sort(divisors.begin(), divisors.end());
        for(int val:divisors)
        {
            cout << val << " ";
        }
    }
};
```

02.) CHECK IF NUMBER IS PRIME OR NOT

CODE 01

```
class Solution{
public:
    int isPrime(int N){
        if (N <= 1) return 0;
        if (N == 2) return 1;
        if (N % 2 == 0) return 0;

        for (int i = 3; i <= sqrt(N); i += 2) {
            if (N % i == 0) return 0;
        }

        return 1;
    }
};
```

MATH FOR DSA

CODE 02

```
class Solution{
public:
    int isPrime(int N){
        if (N <= 1) return 0;
        if (N <= 3) return 1;
        if (N % 2 == 0 || N % 3 == 0) return 0;

        for (int i = 5; i * i <= N; i += 6) {
            if (N % i == 0 || N % (i + 2) == 0) return 0;
        }

        return 1;
    }
};
```

CODE 03

```
class Solution{
public:
    int isPrime(int N){
        if (N <= 1) return 0;
        if (N <= 3) return 1;

        for (int i = 2; i <= sqrt(N); i++) {
            if (N % i == 0) return 0;
        }

        return 1;
    }
};
```

MATH FOR DSA

03.) PRINT PRIME FACTORS

```cpp
#include <bits/stdc++.h>
using namespace std;

void primeFactors(int n)
{
    while (n % 2 == 0)
    {
        cout << 2 << " ";
        n = n / 2;
    }

    for (int i = 3; i <= sqrt(n); i = i + 2)
    {
        while (n % i == 0)
        {
            cout << i << " ";
            n = n / i;
        }
    }

    if (n > 2)
        cout << n << " ";
}

int main()
{
    int n = 315;
    primeFactors(n);
    return 0;
}
```

MATH FOR DSA

04.) SIEVE OF ERATHOTHENES

```cpp
#include <bits/stdc++.h>
using namespace std;

void SieveOfEratosthenes(int n) {
    bool prime[n + 1];
    memset(prime, true, sizeof(prime));

    for (int p = 2; p * p <= n; p++) {
        if (prime[p] == true) {
            for (int i = p * p; i <= n; i += p)
                prime[i] = false;
        }
    }

    for (int p = 2; p <= n; p++)
        if (prime[p])
            cout << p << " ";
}

int main() {
    int n = 30;
    cout << "NUMBERS ARE :- " << n << endl;
    SieveOfEratosthenes(n);
    return 0;
}
```

THANK YOU !