① Introduction

classmate Date ___ Page ___

* **Order of growth**
  constant < $\log\log n$ < $\log n$ < $n^{1/3}$ < $n^{1/2}$ < n < n^2 < n^3 < n^4 < 2^n < n^n

* number of digits in a number $\Rightarrow$ floor ($\log_{10} n + 1$)

* AP $\rightarrow$ n^{th} term $\Rightarrow$ $a+(n-1)d$ Sum $\Rightarrow$ $\frac{n}{2}[2a+(n-1)d]$

* GP $\rightarrow$ n^{th} term $\Rightarrow$ ar^{n-1} * Sum $\Rightarrow$ $\frac{a(r^n-1)}{(r-1)}$ Sum(∞) $\Rightarrow$ $\frac{a}{(r-1)}$

* Sum of even term − Sum of odd term $= \frac{n(d)}{2}$

* $ax^2+bx+c=0$ $\Rightarrow$ $x = \frac{-b \pm \sqrt{D}}{2a}$; $D = b^2-4ac$
  - $\rightarrow D > 0$ real root
  - $\rightarrow D < 0$ imaginary root
  - $\rightarrow D = 0$ equal root

* Median: ordering the data and then take middle value [in case of even length, take mean value of middle & (middle + 1) value].
* Prime number: can be expressed as $(6n+1)$ and $(6n-1)$ {except 2 & 3}
* factors of 24: 1, 2, 3, 4, 6, 8, 12, 24
* LCM (least common multiple) HCF (Heighest Common factor)
* factorial $n! = n*(n-1)*(n-2) \ldots 2*1$ $0! = 1$
* ${}^nP_r = \frac{n!}{(n-r)!}$ ${}^nC_r = \frac{n!}{(n-r)!\,r!}$

* % $\rightarrow$ remainder
* Palindrome $\Rightarrow$ String = reverse (string)

* Trailing zero count in factorial = $\left\lfloor \frac{n}{5} \right\rfloor + \left\lfloor \frac{n}{25} \right\rfloor + \left\lfloor \frac{n}{125} \right\rfloor + \ldots$

* GCD $\leq$ min(a, b)
* Euclidean algorithm:
  if $b<a$ $\Rightarrow$ gcd(a, b) = gcd (a−b, b)

Priyansh

int gcd (int a, int b) {

```
int gcd ( int a, int b) {
        if ( b == 0)
            return a;
        else
            return gcd ( b, a%b);
}
```

TC : log(min(a, b))

* LCM $\geq$ max(a, b)
* a * b = gcd(a, b) * LCM(a, b)

Prime number (to check if given number n is prime or not)

```
bool isprime ( int n) {
    if ( n==1)
        return false;     //neither prime nor composite
    if (n==2 or n==3)
        return true;
    if ( n%2==0 or n%3==0)
        return false;
    for ( int i=5; i<= sqrt(n); i+=6 )
    {
        if ( n%i ==0 or n% (i+2) ==0)
            return false;
    }
    return true;
}
```

TC : $\Theta(\sqrt{n})$

based on fact that Prime no. can be represented as (6n-1) & (6n+1)

Prime factors (print all prime factors of given number n)

```
void printprimefactors ( int n) {
    if (n <= 1)
        return;
    while ( n%2 == 0) {
        print(2);
        n/=2;
    }
```

Priyansh

```
while (n % 3 == 0) {
        print(3);
        n /= 3;
}
for (int i = 5; i <= sqrt(n); i += 6) {
        while( n % i == 0) {
            print(i);
            n /= i;
        }
        while ( n % (i+2) == 0) {
            print (i+2);
            n /= (i+2);
        }
}
if (n > 3)
        print(n);
}
```

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

Sample Input : n = 24   Sample output : 2, 2, 2, 3

All divisor of a number (to ~~check~~ print all divisors of a given no.)

```
void printdivisor (int n) {
    for( int i=1; i <= sqrt(n); i++ ) {
        if( n % i == 0 and i*i != n )
            print(i);
    }
    for( int i = sqrt(n); i >= 1; i-- ) {
        if( n % i == 0)
            print( n/i );
    }
}
```

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

Sample Input : n = 24   Sample output : 1, 2, 3, 4, 6, 8, 12, 24

n = 25   : 1, 5, 25

Priyansh

Sieve of Erathosthenes (to print all prime no. <= given no. (n))

```
void sieve (int n) {
    vector <bool> isprime (n+1, true);
    for (int i=2; i<=n; i++) {
        if (isprime [i]) {
            cout<<i<<endl;
            for (int j=i*i; j<=n; j+=i) {
                isprime [j] = false;
            }
        }
    }
}
```

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

Pow(x, n) [to return integer = x^n]

```
int power (int x, int n) {
    if (n == 0)
        return 1;
    int temp = power (x, n/2);
    temp *= temp;
    if (n%2 == 0)
        return temp;
    else
        return temp*x;
}
```

TC: $O(\log n)$

Auxilary Space: $O(\log n)$

Every number can be written as a sum of power of 2.

bits(x) [to go through all bits of a given number]

```
int bits (int n) {
    while (n>0) {
        if (n%2 == 0)
            // bits is 0
        else
            // bits is 1
```

TC: $O(\log n)$

Priyansh

eg:
10: 1010
19: [redacted] 10011

18/2/24


```
        n /= 2;
    }
}
```

Iterative power [find power x^n given % m]

```
int powermod( int x, int n, int m ) {
    int res = 1;
    while ( n>0 ) {
        if ( n%2 == 0 ) {
            res = (res*x) % m;
        }
        x = (x*x) % m;
        n = n/2;
    }
    return res;
}
```

TC : $\Theta(\log n)$

Auxiliary Space : $\Theta(1)$

Modular Arithmetic

$(a+b)\%m = [(a\%m) + (b\%m)]\%m$

$(a-b)\%m = [(a\%m) - (b\%m)]\%m$

$(a*b)\%m = [(a\%m) * (b\%m)]\%m$

Digits in a factorial

$n! = n*(n-1)*(n-2)*(n-3) \dots *4*3*2*1$

$\Rightarrow \log(n!) = \log n + \log(n-1) + \log(n-2) \dots + \log 3 + \log 2 + \log 1$

$\therefore$ digits in $n!$ $\Rightarrow \boxed{\text{floor}(\log(n!)) + 1}$

$$\boxed{\text{floor}(\log n + \log(n-1) + \log(n-2) + \dots + \log 2 + \log 1) + 1}$$

Priyansh