

### Digit Extraction

$N = 7789$

$$\begin{aligned} &= 7789 \% 10 = 9 \\ \xrightarrow{1/10} & 7789 \% 10 = 8 \\ \xrightarrow{1/10} & 77 \% 10 = 7 \\ \xrightarrow{1/10} & 7 \% 10 = 7 \\ \xrightarrow{1/10} & 0 \end{aligned}$$

while ( $N > 0$ )  
 {  
 Last digit + E =  $N \% 10$ ,  
 N =  $N / 10$ ;  
 }  
 code 1  
 If problem solved by  
 division →  
 $T.C = O(\log_{10}(n))$

To count digits:  
 $T.C = O(\log_{10}(n))$   
 declare count var = 0;

cnt = 0;  
 while ( $n > 0$ )  
 {  
 td =  $n \% 10$ ;  
 cnt = cnt + 1;  
 N =  $N / 10$ ;  
 }  
 code 2

number of time its  
 divisible by 10.

int cnt = (int)  $\log_{10}(n) + 1$ ;

If the division is happening by 10. → log(10)  
 for time comp.

### Reverse Number

RevN = 0

while ( $n > 0$ ) {

LD =  $n \% 10$ ;

N =  $N / 10$ ;

RevN = (RevN  $\times 10$ ) + LD ;

}

## \* Palindrome

$121 \rightarrow$  Reverse  $121 \rightarrow 121$   $\rightarrow$  concept  
 Number's reverse is same

$$\boxed{\text{RevN} = N}$$

$\Rightarrow$  Make another variable named duplicate and store N there  
 Lastly if  $\text{Dup} = \text{RevN}$  then the number is palindrome.

```
int n;
cin >> n;
```

```
int revN = 0, dup = n;
```

```
while (n > 0) { int ld = n % 10;
  revN = (revN * 10) + ld;
```

```
n = n / 10;
```

} if  $\text{revN} = \text{dup}$  print true  
 else false

Date: / /  
 Sat  Sun  Mon  Tue  wed  Thu  Fri

Theme:

### ① Armstrong Number

$$N = 371 = 3^3 + 7^3 + 1^3$$

```
sum = 0, dup = n;
while (n > 0)
{ d = n % 10;
  sum = sum + (d)^3;
  n = n / 10;
}
```

```
if dup == sum "Arm"
else "No"
```

### ② Print all divisors

$$N = 10 = 1, 2, 5, 10$$

```
for (int i = 1; i <= n; i++)
{ if (n % i == 0)
    cout << i }
```

### ③ vector<int> ls;

```
for (int i = 1; i <= sqrt(n))
{ if (n % i == 0)
    ls.push_back(i);
  if ((n / i) != i)
    ls.push_back(n / i); }
```

$O(n \log n)$ :  $n = \frac{\text{num of factors}}{\sqrt{n}}$

```
sort(ls.begin(), ls.end());
for (auto it : ls) cout << it
<< "
```

→ Also can write  $i * i \leq n$

$\rightarrow +c \rightarrow O(\sqrt{n})$

Theme:

## \* Prime Numbers

→ exactly 2 factors, 1 and itself

```

int cnt = 0;
for (int i=1; i<n; i++) {
    if (n % i == 0)
        cnt++;
}
if (cnt == 2) prime
else not prime
    
```

→ Big O(N)

```

int cnt = 0;
for (int i=1; i<=n; i++) {
    if (n % i == 0) {
        if ((n/i) != i)
            cnt++;
    }
}
if (cnt == 2) cout << true
else cout << false
    
```

→ Big O(sqrt(n))

## \* GCD / HCF

$$N1 = 9 \quad N2 = 12$$

$$\begin{array}{l} \downarrow \\ 1, 3, 9 \end{array} \quad \begin{array}{l} \downarrow \\ 1, 2, 3, 4, 6, 12 \end{array}$$

$$\begin{array}{l} \xrightarrow{\text{GCD}} 3 \\ \xrightarrow{\text{HCF}} 3 \end{array}$$

Euclidean Algo → gcd(a, b) → gcd(a - b, b) → 0  
 $a > b$   
 $\gcd(a, b) = \gcd(a \% b, b)$

(greater ∵ smaller)

a, b  
 while ( $a > 0$  &  $b > 0$ )

    if ( $a > b$ )    $a = a \% b$ ;   if one  $i > 0$   
     else            $b = b \% a$ ;   other  $i = \gcd$

    if ( $a == 0$ ) print(b)  
     else print(a)

TC → O(log<sub>2</sub>(min(a, b)))

Theme:

Date: / /  
Sat Sun Mon Tue wed Thu Fri

code:

```
while (a>0 && b>0)
    if (a>b) a = a % b;
    else b = b % a;

    if (a==0) return b;
    else return a;
```