

* Digit Extraction

$N = 7789$
 \swarrow
 $= 7789 \% 10 = 9$
 \swarrow
 $778 \% 10 = 8$
 \swarrow
 $77 \% 10 = 7$
 \swarrow
 $7 \% 10 = 7$
 \swarrow
 0

```

while (N > 0)
{
    last digit = N % 10;
    print();
    N = N / 10;
}
    
```

code 1

* If problem solved by division \rightarrow
 $T.C = O(\log_{10}(N))$

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

```

cnt = 0;
while (n > 0)
{
    ld = n % 10; cnt = cnt + 1;
    N = N / 10;
}
    
```

code 2

number of times its
 divisible by 10.

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

if the division is happening by 10 $\rightarrow \log(10)$
 for time comp.

* Reverse Number

RevN = 0

```

while (n > 0)
{
    LD = N % 10;
    N = N / 10;
    RevN = (RevN * 10) + LD;
}
    
```

Theme:

Date: / /

Sat Sun Mon Tue Wed Thu Fri

* Palindrome

121 \rightarrow Reverse 121 \rightarrow 121

Number's reverse is same

concept

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

\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 (revN == dup) true
```

```
else false
```

Palindrome

Theme:

Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

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

```
sum = 0, dup = n;
while (n > 0)
{
    ld = n % 10;
    sum = sum + (ld3);
    n = n / 10;
}
if (dup == sum) "Arm"
else "No"
```

② Print all divisors
 $N = 10 = 1, 2, 5, 10$

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

③
 simple

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

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

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

* Also can write $i * i \leq n$
 \rightarrow to $\rightarrow O(\sqrt{n})$

Theme:

* Prime Numbers

→ exactly 2 factors, 1 and itself

```

ent = 0;
for (i = 1; i <= n; i++)
    if (n % i == 0)
        ent++;
    }

```

if (ent == 2) prime
 else not prime

→ Big O(N)

```

int ent = 0;
for (i = 1; i <= n; i++)
    if (n % i == 0) {
        ent++;
        if (n / i != i) ent++;
    }
    }

```

if (ent == 2) cout << true
 else cout << false

→ Big O(sqrt(n))

* GCD / HCF

N1 = 9 N2 = 12

↓
 1, 3, 9

↓
 1, 2, 3, 4, 6, 12

GCD → 3
 → HCF → 3

Eucledian 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;
 else b = b % a

↓
 if one is 0
 other is gcd

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

TC → O(log(min(a, b)))
 changing

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;
```