Math Basics

Digit Concept

```
int n = 7789 %10 = 9
   /10↓ 778.9 %10 = 8
      /10↓ 77 %10 = 7
         /10↓ 7 %10 = 7
            /10↓ 0
```

extraction of digits is taking input 'n' and while n>0, and prints the last digit % 10. Then divides n by 10 to iterate till 0.* IN REVERSE ORDER

Count Digits

```
count = 0;
while(n > 0){
    lastdigit = n %10;
    count += 1;
    n = n /10;
}
```

* just make a counter for the coded loop above

↑

* time complexity is $O(\log_{10}(N))$ b/c loop is getting run number of times it is divided by 10

* If divison is happening by 10, $\log_{10}(N)$. by 2, $\log_2(N)$

Reverse a Number

n = 7789 → 9877

```
reverseNum = 0;
lastDigit = n %10;
reverseNum = (reverseNum x10) + lastDigit;
}
return reverseNum;
```

Digit Concept

Day 9

2/19/25

Check Palindrome

palindrome is ex 121 == 121, 12321 === 12321

take reverse number and compare w/ original

Armstrong Number

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

make sum = 0 variable

sum = sum + (lastDigit ** 3);

run rest of code the same

Print All Divisors O(N)

36 → 1, 2, 3, 4, 6, 9, 12, 18, 36 // divide w/o remainder

loop from 1 → n

check it leaves 0 remainder

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

Sorting

first check if 'i' is factor

also check if the factor is not equal to i, so no duplicate

store in a list vector<int> ls;

sort list for(auto it: ls) cout << it << " ";

tc is O(# of factors → $\log_2$)

Check For Prime O(N) Brute Force

*check if factors are 1 & itself

```
int cnt = 0
for(i=1; i<=n, i++){
  if(n%i==0) cnt++;
  if(cnt==2) ✓;
  else X;
```

```
if(n%i==0)
  cnt++;
if((n/i) != i)
  cnt++;
if(cnt==2) ✓;
else X;
```

2/20/25

GCD of HCF

N1=9 N2=12

1, 3, 9 → highest divisor

1, 2, 6, 12, 3, 4 so gcd(9,12) = 3

From 1 → n2, check every number to see if it divides both, keep replacing gcd till highest number

```
for(i=1; i<= min(n1,n2); i++){
    if(n1 % i == 0 && n2 % i == 0){
        gcd = i;
}}
```

TC = O(min(n1, n2))

flipped ↓

```
for(i = min(n1,n2); i >= 1 = i--){
    if(n1 % i == 0 && n2 % i == 0){
        print i;
        Break; // same time complexity
}}
```

EQUILATERAL ALGORITHM

gcd(a,b) = gcd(a-b, b)

gcd(10,5) = gcd(5, 5) → ged(5-5, 5) (0, 5)

so when a > b

gcd(a,b) = gcd(a % b, b)

* if one value = 0, the other value is the GCF