Todays Content:

- GCD Intro
- Properties of GCD
- GCD function
- GCD Problems
  - a) Check Subsequence with gcd = 1
  - b) Delete 1 element such that gcd of remaining elements in your array is max
  - c) Pubg : Assignment

GCD : greatest common divisor / HCF → highest common factor

$gcd(a, b) = x : \{ x \text{ is greatest number s.t } a \% x == b \% x == 0 \}$

$gcd(15, 25) : 5$

1	1
3	5
5	
15	25

$gcd(12, 30) : 6$

1	1
2	2
3	3
4	5
6	6
12	10
	15
	30

$gcd(10, -25) : 5$

1	-25
2	-5
5	-1
10	1
	5
	25

$gcd(0, 8) : 8$

1	1
2	2
3	4
⋮	8
8	
9	
⋮	
∞	

$gcd(0, -10) : 10$

1	-10
2	-5
3	-2
⋮	-1
10	1
⋮	2
∞	5
	10

$gcd(-16, -24) : 8$

1	1
2	2
4	3
8	4
16	6
	8
	12
	24

$gcd(-2, -3)$

+ve

1	1
2	3

$\boxed{gcd(0, 0) : \text{not defined}}$

+ve

1	1
2	2
3	3
⋮	⋮
∞	∞

Properties of gcd(a, b):

- $\gcd(a, b) = \gcd(b, a)$] Commutative Property
- $\gcd(a, b) = \gcd(|a|, |b|)$ // Sign doesn't matter
- $\gcd(0, n) = |n|$ // if $n != 0$

- $\gcd(a, b, c) = \gcd(\gcd(a, b), c)$
  $= \gcd(\gcd(b, c), a)$
  $= \gcd(\gcd(a, c), b)$] Associative property

Special Property: [A, B > 0 && A > B]

Ass: $\gcd(A, B) = n \Rightarrow$ n is greatest number such that $A \% n == B \% n == 0$

$\gcd(A - B, B) = n$

if $(A - B) \% n == 0$ ✓ && $(B \% n) == 0$ ✓

%enpad

$= [\underbrace{A \% n}_{0} - \underbrace{B \% n}_{0} + n] \% n$

$= n \% n = 0$

Claim: $A > B$ & $A, B > 0$

$$\boxed{\gcd(A, B) = \gcd(A - B, B)}$$

$\gcd(\overset{a}{23}, \overset{b}{5}) = \gcd(\overset{a}{18}, \overset{b}{5}) = \gcd(\overset{a}{13}, \overset{b}{5}) = \gcd(\overset{a}{8}, \overset{b}{5}) = \gcd(\overset{a}{3}, \overset{b}{5})$

$\gcd(23, 5) = \gcd(3, 5) = \gcd(23\%5, 5)$

$\boxed{A\%B = A - \{\text{Greatest mul of } B <= A\}}$

$A >= B$

$\gcd(A, B) \longrightarrow \gcd(A-B, B)$

$\rightarrow \gcd(A-2B, B)$

$\rightarrow \gcd(A-3B, B)$

$\rightarrow \gcd(A-4B, B)$

$\vdots$

$\rightarrow \gcd(A-xB, B)$

↳ max multiple of B we can remove from A

$\boxed{\gcd(\underset{}{A, B})_{x>=y} = \gcd(A\%B, B)_{x<y}}$ → <u>Case I</u>

$\boxed{\gcd(A, B)_{x>=y} = \gcd(B, A\%B)_{x>=y}}$ → <u>Case-II</u>

```
int gcd(a, b) {   // It won't work
    if(b==0) {return a}
    return gcd(a%b, b)
}
```

gcd(24, 16)

↓

gcd(8, 16)

↓

gcd(8, 16)

↓

gcd(8, 16)

↓

⋮

∞ loop

Memory limit Exceed

Recursion - 1

Doubts Session

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

→ main() {

gcd(|a|, |b|)

}

Both of the numbers will be non negative

gcd(24, 16)

↓

gcd(16, 8)

↓

gcd(8, 0):

↓

gcd(0 8%0)

gcd(8 16)

↓

gcd(16, 8)

↓

gcd(8 0)

ans = 8

gcd(10, 10)

↓

gcd(10, 0) : ans = 10

gcd(0, 8)

↓

gcd(8, 0) : ans = 8

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

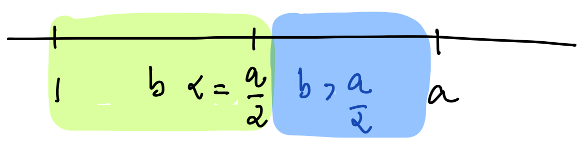
TC: $O\left[\log_2^{\max(a,b)}\right]$?

SC: $O\left[\log_2^{\max(a,b)}\right]$

When log?

$a \to a/2 \to a/4 \cdots 1$

if$(a >= b)$: $gcd(a, b) = gcd(a\%b, b)$



Case I: $b <= \frac{a}{2}$

↳ $\underbrace{a\%b} < b <= \frac{a}{2}$

$\boxed{a\%b < \frac{a}{2}}$

$gcd(a, b) = gcd(\overset{a\%b}{<\frac{a}{2}}, b)$

After 1 iteration a become half

After $\log_2^a$ code breaks

Case-II: $b > \frac{a}{2}$

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

$gcd(a, b) = gcd(<\frac{a}{2}, b)$

After $\log_2^a$ code break

Case II: $b > \frac{a}{2}$

$2b > a$

$2b - a > 0$

$a - 2b < 0$

Add a in both

$2a - 2b < a$

$2(a-b) < a$

$a - b < \frac{a}{2}$

$= a - 2b < \frac{a}{2}$

$= a - 3b < \frac{a}{2}$

$= a - nb < \frac{a}{2}$

$= a\%b < \frac{a}{2}$

1Q) Given ar[N] calulate gcd of entire array

ar[3] = { 6 12 15 } : ans = 3

6 3 → ans = 3

ar[4] = { 8 16 12 10 } : ans = 2

8 4 = 2

```
int gcdarr( int ar[], int n) {
    ans = 0 ?
    i = 0; i < n; i++) {
        ans = gcd (ans, ar[i])
    }
    return ans;
}
```

TC: $n \log_2 (\max \text{ of } ar[])$

$TC: O(N + \log_2 ar[0]) = O(N)$

Doubts Session ?

Dry run:

Dry run: ar[4] =

0	1	2	3
8	16	12	10

ans = 0 8 8 4 2

i=0: ans = gcd (ans, ar[0]) ⇒ gcd (0, 8) = 8

i=1: ans = gcd (ans, ar[1]) ⇒ gcd (8, 16) = 8

i=2: ans = gcd (ans, ar[2]) ⇒ gcd (8, 12) = 4

i=3: ans = gcd (ans, ar[3]) ⇒ gcd (4, 10) = 2

obs: In our ans = gcd (ans, ar[i])

// Say ans is continously decreasing → $\log_2 ar[0] + N$

ans = ar[0] .... → 1 → TC: O(N)

$\log_2 ar[0]$ iteratn

20) Given an array, check if there exists a subsequence with gcd = 1
↳ boolean

ar[5] = {4 6 8 8}, seq = {4, 8}

ar[5] = {16 10 6 15 27}, seq = {10, 6, 15}

ar[4] = {6 12 3 18}, seq = { }

Idea1: For every subsequence, get gcd of entire subseq = 1

→ TC: $2^n * O(N) \Rightarrow O(2^N * N)$

Idea2:

Say 7 elements:

↳ A B C D E F G

Say there exists a subsequence with GCD = 1

gcd(B, D, E, G) = 1

gcd (A B C D E F G) =

= gcd (gcd(A, C, F), gcd(B, D, E, G))

= gcd(gcd(A, C, F), 1)

= 1

Claim: if there exist a subsequence with gcd == 1

⇒ GCD of entire array will also = 1

obs: GCD of entire array = 1 : Subsequence there {return T}
≠ 1 : Subsequence not {return F}

TC: O(N)

Deleteone:

Given ar[N] elements, we have to delete 1 Element, such that gcd of remaining array is max
↳ max gcd

Ex:

0	1	2	3	4	
24	16	18	30	15	= 1
*	gcd of all elem				

0	1	2	3	4	
24	16	18	30	15	= 3 :
[ ]	*	[		]	

0	1	2	3	4	
24	16	18	30	15	= 1
[	]	*	[	]	

0	1	2	3	4	
24	16	18	30	15	= 1
[		]	*	[ ]	

0	1	2	3	4	
24	16	18	30	15	= 2
[			]	*	

Idea: i = 0; i < n; i++) {

Exclude ar[i] & get of all elements,

}

Finally overal max

TC: O(N * N) = O(N²)

```
int deleteOne(int ar[], int n){

    ans = 0
    int pfgcd[n]; -> TC : O(N) TODO
        // pfgcd[i] = gcd of all elements [0 i]

    int sfgcd[n]; -> TC : O(N) TODO
        // sfgcd[i] = gcd of all elements [i, n-1]

    // Edge Case - I : i = 0, we need to get of all [1, N-1] = sfgcd[1]
    // Edge Case - II : i == n-1, we get to get of all [0, N-2] = pfgcd[N-2]
    ans = max( sfgcd[1], pfgcd[N-2])
    i = 1 ; i < n-1 ; i++){
        // gcd of all elements [0 n-1] excluding i^th ele
        // -> gcd { gcd[0, i-1], gcd [i+1, N-1] }
        left  = gcd [0, i-1] = pfgcd [i-1], Edge i==0
        right = gcd [i+1, N-1] = sfgcd [i+1], Edge i==n-1
        ans = max (ans, gcd(left, right))
    }
    return ans;
}
```

<u>TC</u>: O(N + N + N) -> O(N)

<u>SC</u>: O(N + N) -> O(N)

→ TODO

```
int ar[N],
int pfgcd[N]
ans = 0;
i = 0; i < n; i++){
    ans = gcd(ans, ar[i])
    pfgcd[i] = ans
}
```

```
int sfgcd[N]
ans = 0
i = n-1; i >= 0; i--){
    ans = gcd(ans, ar[i])
    sfgcd[i] = ans
}
```

Doubts:

→ GCD of arr[] :

#no: of iterations : 1

$gcd(1, a) = a$

$a[N] :\quad a_0 \quad a_1 \quad a_2 \quad a_3 \; \ldots \; a_n \quad a_{n+1} \; \ldots\ldots \; a_{N-1}$

$ans = 0 \rightarrow a_0 \quad g_1 \quad g_2 \quad g_3 \quad g_n = 1$

Iterations

Step : 1 :	$ans = gcd(ans, a_1) = g_1$	t_1
Step : 2 :	$ans = gcd(ans, a_2) = g_2$	t_2
Step : 3 :	$ans = gcd(ans, a_3) = g_3$	t_3
Step : n :	$ans = gcd(ans, a_n) = 1$	t_n

$\log_2^{a_0}$ iterations

$ans = 100$

Step1 : 1	$100 \longrightarrow 25$:	2	Total : 6 = $\log_2^{100}$
Step2 :	$25 \longrightarrow 3$:	3	
Step3 :	$3 \longrightarrow 1$:	1	

TC: SC ⟶

redn → 2n → O(1)

Brief: 80,000 records ⟶ 10kB → 300GB

: +2 years: 300GB