# Lecture : Maths: GCD

## Agenda.

- GCD Basics
- GCD properties
- Euclid algorithm.
- Maximise GCD
- PUBG

Class starts at 7:05 AM

# [HCf] GCD: Greatest common divisor

gcd(12, 30)   ans = 6

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

gcd(15, 25) = 5

gcd(15, −25) = 5 [15 % 5 == 0 && −25 % 5 == 0]

−5 [15 % −5 == 0 && −25 % −5 == 0]

gcd(a, b) = +ve no.

gcd(a, b)

Min value   1.   gcd(5, 11) = 1

Max value   min(a, b)   gcd(6, 48) = 6

Approach 1

```
for( i = min(a, b);  i >= 0;  i-- ) {

        if ( a % i == 0  &&  b % i == 0 ) {

              return i;

        }
}
```

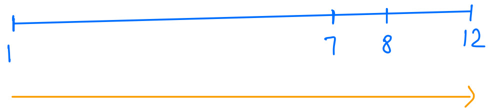
TC: o(min(a, b)) ≃ o(n)

SC: o(1)

## Approach2

gcd ( 12 , 30 )

gcd =



$O(\min(a, b))$

find factors of 12 [ 1, 2, 3, 4, 6, 12 ]

find factors of smaller no.

for every factors: (f)

check $a \% f == 0$ && $b \% f == 0$

TC: $O(\sqrt{\min(a, b)}$

SC: Think about it

## Properties

1. $gcd(a, b) = gcd(b, a)$

2. $gcd(1, a) = 1.$ $\Big[$ Range $1, min(a, b)$

$1, min(1, a)$

$1, 1 \Big]$

3. $gcd(0, a) = 0$ $[0 \% 0 == 0 \ \&\& \ a \% 0 == 0]$ ✗

undefined

$a$ $[0 \% a = 0 \ \&\& \ a \% a = 0]$ ✓

4. $gcd(a, b, c, d) \ldots = gcd(gcd(gcd(a, b), c), d) \ldots\ldots$

$gcd(8632, 8650)$

$\downarrow$

tough to find

* $gcd(a, a) = a$

5. $gcd(a, b) = gcd(a, b-a)$ $b > a$

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

Eg: $gcd(15, 25) = 5$

$gcd(\overset{a}{15}, \overset{b}{25}) = gcd(15, \overset{10}{25-15}) \longrightarrow gcd(15, 10)$

$= gcd(\overset{5}{15-10}, 10)$

$= gcd(5, \overset{5}{10-5})$

$= gcd(5, 5) \longrightarrow 5$

$\gcd(8632, 8650) = \gcd(8632, 8650 - 8632)$

$= \gcd(8632, \underset{\uparrow}{18})$

factors of 18 [1, 2, 3, 6, 9, 18]

$\gcd(120, 270) = \gcd(120, \overset{150}{270 - 120})$

$\gcd(120, 150)$

$\gcd(120, 30)$

$\gcd(\overset{90}{120 - 30}, 30)$

$\gcd(60, 30)$

$\gcd(30, 30) = \boxed{30}$

$\uparrow \quad \uparrow$

$a = b$

$a = 0, \quad b$

$b = 0, \quad a$

↓ smaller problems

$\gcd(-12, -18) = \gcd(-12 - (-18), -18)$

$= \gcd(6, -18)$

$= \gcd(6 - (-18), -18)$

$= \gcd(24, -18)$

$= \gcd(42, -18)$

$= \gcd(60, -18)$

$= \gcd(78, -18)$

$\vdots$

if no are -ve −

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

**Proof:** $\gcd(a, b) = \gcd(a, b-a)$ $b > a$

$\gcd(a, b) = x$ $\gcd(a, b-a) = y$

To prove: $x = y$

$\gcd(a, b) = x$

$\downarrow$

$a \% x = 0$ — ①

$b \% x = 0$ — ⑪

⑪ — ①

$b \% x - a \% x = 0$

$(b-a) \% x = 0$

$x$ divides $b-a$.

$x$ is divisor of $a$ & $b-a$

$x <= y$

$\gcd(a, b-a) = y$

$\downarrow$

$a \% y = 0$

$(b-a) \% y = 0$

$(b \% y - \underset{\overset{\Downarrow}{0}}{a \% y} + y) \% y = 0$

$(b \% y + y) \% y = 0$

$b \% y = 0$

$\downarrow$

$y$ divides $b$

$y$ is divisor of $a$ & $b$

$y <= x$

$x == y$

$\gcd(1200, 3) = \gcd(1197, 3)$ $-3$

$\gcd(1194, 3)$ $-3$

$\gcd(1191, 3)$ $-3$

$\gcd(1188, 3)$ $-3$

$\gcd(1185, 3)$ $-3$

$\gcd(1182, 3)$ $-3$

$\gcd(1179, 3)$

⋮

$\gcd(0, \_)$ (or) $\gcd(\_, 0)$

repetitive subtraction

[ Division ]

$21 \div 3 = \xrightarrow{\text{steps}} 21 - 3 - 3 - 3 - 3 - 3 - 3 - 3 - \_\_$ $[0, 1, 2]$

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

$\hookrightarrow b - a - a - a - a - a - \dots$

left with $< a$

$[0, a-1]$

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

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

$\gcd(1200, 3) =$

$\gcd(1197, 3)$ $-3$

$\gcd(1194, 3)$ $-3$

$\gcd(1191, 3)$ $-3$

$\gcd(1188, 3)$ $-3$

$\gcd(1185, 3)$ $-3$

$\gcd(1182, 3)$ $-3$

$\gcd(1179, 3)$

repetitive subtraction

[ Division ]

⋮

$\gcd(0, \_)$ (or) $\gcd(\_, 0)$

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

$\gcd(1200, 3) = \gcd(1200 \% 3, 3)$

$= \gcd(0, 3) = 3$

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

TC: $O(\log_2 \min(a,b))$

$gcd(5, 11)$

$gcd(5, \overset{1}{11\%5})$

$gcd(5, 1)$

$gcd(\overset{0}{5\%1}, 1)$

$gcd(0, 1)$

↑

return 1.

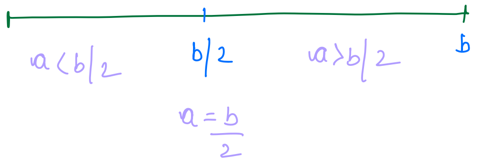
## TC of GCD: Euclid algo: $\gcd(a,b) = \gcd(a, b\%a)$

$\gcd(a, b)$

$\downarrow$ $a > \frac{b}{2}$, $b = b/2$.

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

$b > a$ (or) $a < b$



$\downarrow$

$\log_2 (\quad)$

$a < b/2$

$a = b/2$

$\gcd(a, b)$

$\uparrow$ $b/2$

$\downarrow$ $b\% b/2$

$\gcd(b/2, 0)$

$O(1)$

$[\gcd(9,18)]$

$\downarrow$

$\gcd(9,0) = 9$

$a > b/2$

$a > b/2$

$2a > b$

$0 > b - 2a$

Add b both sides

$b > 2b - 2a$

$b > 2(b-a)$

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

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

$\uparrow$ $b > a$

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

$b = 8, a = 5$
$b = 12, a = 9$
$b = 118, a = 62$

Break: 8:48 AM

# PUBG

* n players, each has strength A[i].

ith $\xrightarrow{\text{attacks}}$ jth player $\Big\{$ jth player: → max(0, A[j] - A[i])

↓

| ith | jth | | |
|---|---|---|---|
| 2 | 8 | → | 8 - 2 = 6 |
| 8 | 6 | → | 6 - 8 = ~~-2~~ 0 |

When strength of any player reaches 0, it loses the game until 1 survivour remains

Tell the minimum health of last surviving person?

Eg: arr[] = $\{ \overset{0}{6}, \overset{1}{4} \}$

gcd(2) → ans = 2

0th player attacks 1st player

0th = 6 ✓

1st = 4 - 6 = ~~-2~~ 0

1st player attacks 0th player

0th player = 6-4=2

1st player = 4.

0th player attacks 1st player

0th player = 2

1st player = 4-2 = 2

0th player attacks 1st player

0th player = 2 ✓

1st player = 2-2 = 0

**Eg2:** arr[] = [ $\overset{0}{2}$ , $\overset{1}{3}$ , $\overset{2}{4}$ ] gcd = 1

2nd player attacks 1st player
0th = 2
1st = 3−4 = $-\cancel{1}$ 0
2nd = 4.

2nd player attacks 0th player
0th = 2−4 = $-\cancel{2}$ 0
2nd = 4.

Last: strength = 4.

0th player attacks 2nd player
arr = [ 2, 3, 2 ]

0th player attacks 2nd player
arr = [ 2, 3, 0 ]

0th player attacks 1st player
arr = [ 2, 1, 0 ]

1st player attacks 0th player
arr = [ 1, 1, 0 ]

1st player attacks 0th player
arr = [0, 1, 0]

Last: strength = 1 ✓

**Observation** for min health of last man standing –
smaller strength should attack higher strength.

arr[] = [ $\overset{0}{3}$, $\overset{1}{8}$, $\overset{2}{5}$, $\overset{3}{11}$, $\overset{4}{13}$ ] ≡ gcd of array

attack 2 times

3 5

3 2 5 11 13

3 2 8 10

3 2 5 7

3 2 2 4

3 2 2 1

2 times

2 times

3 times

0 0 0 1

```
int minstrength(int[] arr){
        int ans = 0;
o(n) —— for(i : arr){
            ans = gcd(ans, i);
                  ↑
                  log2 min(a,b)
        }
    return ans;
}
```

TC: $o(n \log_2 \min(a,b)) \simeq o(n \log_2 n)$

SC: $o(1)$

**Qu** Given arr[n].

Delete one el such that GCD is maximum.

arr[] = [ 12  15  18 ]

**case1:** 12 is deleted

arr: 15  18 , gcd = 3

**case2** 15 is deleted

arr: [ 12 , 18 ]  gcd = 6  Ans

**case3** 18 is deleted

arr: [12  15]  gcd = 3

**Brute force:**

```
ans = 0;
for (i = 0; i < n; i++) {   —— o(n)
    curr = 0
    for (j = 0; j < n; j++) {   — o(n)
        if (i != j) {
            curr = gcd(curr, arr[j]);   —— o(log2 min(a,b))
        }
    }
    ans = max(ans, curr);
}
```

TC: $n^2 \log n$.

SC: o(1)

## Approach2

$$\text{arr}: [\ \underbrace{a\quad b\quad c}_{gcdleft}\quad d\quad \underbrace{e\quad f\quad g\quad h\quad i}_{gcdright}\ ]$$

$$gcd_{\text{deleting } d} = gcd(gcdleft, gcdright)$$

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  | 0 | 1 | 2 | 3 | 4 |
| arr: | 12 | 15 | 24 | 18 | 30 |
| pf: | 12 | 3 | 3 | 3 | 3 |
| sf: | 3 | 3 | 6 | 6 | 30 |

pf[i] = gcd(pf[i-1], arr[i])

gcd = sf[1]

gcd = gcd(pf[0], sf[2])

gcd(pf[1], sf[3])

**Generalise**

Deleting ith element –

gcd = gcd(pf[i-1], sf[i+1])

Edge cases:

i = 0

gcd = gcd(pf[-1], sf[1])

gcd = sf[1].

i = n-1

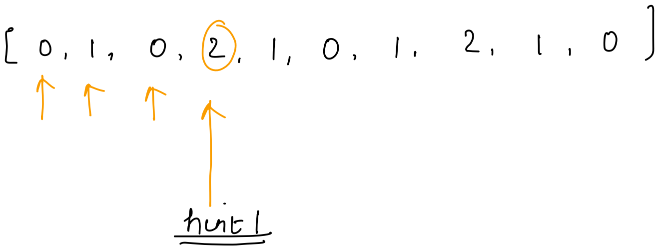
gcd = pf[n-2]

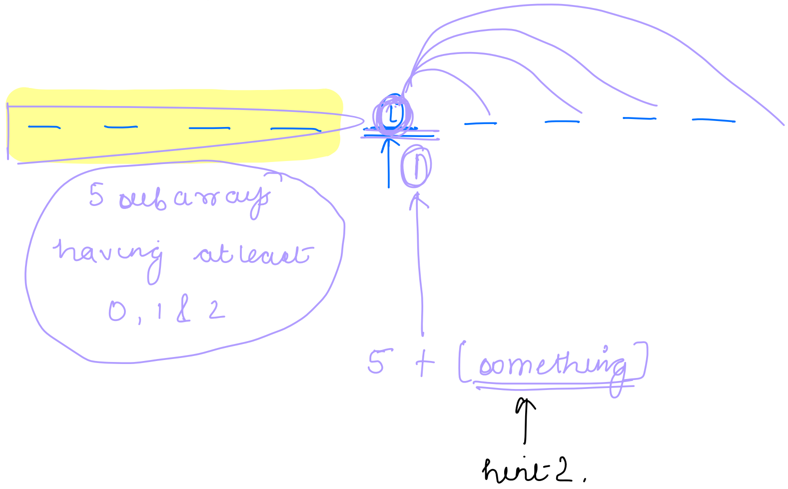
TC: O(n logn)

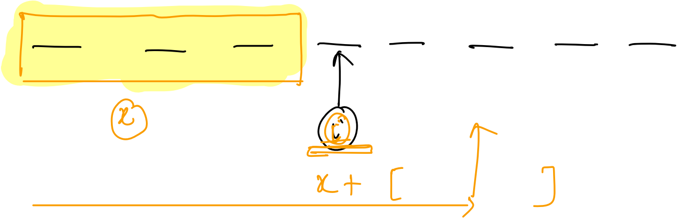
SC: O(n)

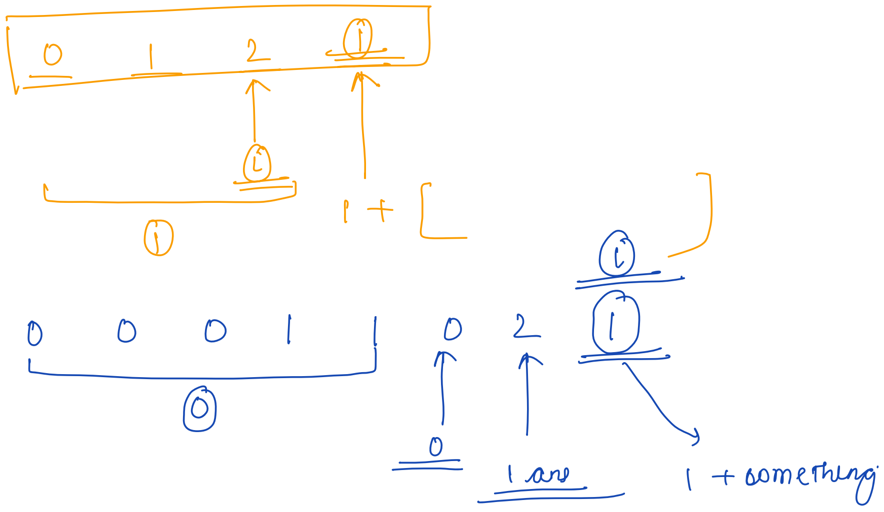
Thankyou ☺

# Doubts:


[ 0, 1, 0, 2, 1, 0, 1, 2, 1, 0 ]
hint 1


i
5 subarrays having atleast 0, 1 & 2
i
5 + [something]
hint 2.


x
i
x + [ ]


0 1 2 i
i
1
1 + [ ]
i
0 0 0 1 1 0 2 i
0
0
1 ans
1 + something