Today's agenda

- GCD Intro
- Properties of gcd
- gcd function
- gcd Problems
  - Subsequence with gcd = 1
  - Delete 1 element.

**GCD:** greatest common divisor / HCF → Highest common factor.

gcd(a,b) : x { x is greatest number, a%x==0 && b%x==0 }

gcd(15,25)

| 15 | 25 |
|---|---|
| 1 | 1 |
| 3 | 5 |
| 5 | 25 |
| 15 | |

gcd(12,30)

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

gcd(10,-25)

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

gcd will always be +ve.

gcd(0,8)

| 0 | 8 |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 4 |
| 4 | 8 |
| 8 | |
| ∞ | |

gcd(0,-10) = gcd(0,10)

| 0 | 10 |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 5 |
| : | 10 |
| 10 | |
| : | |
| ∞ | |

gcd(-16,-24)

gcd(16,24)

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

gcd(-2,-3)

gcd(2,3)

| 2 | 3 |
|---|---|
| 1 | 1 |
| 2 | 3 |

gcd(0,0): Not defined

| 0 | 0 |
|---|---|
| 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|)
- gcd(0,n) = |n|, if n != 0
- gcd(a,b,c) = gcd(gcd(a,b),c) } Associative Property

→ Given 2 numbers a & b find gcd of them.

b GCD

$a < b$

$gcd(24, 34)$

$$\begin{array}{r} 1 \\ 24 \overline{)\ 34} \\ \underline{24} \\ 10 \overline{)\ 24} \\ \underline{20} \\ 4 \overline{)\ 10} \\ \underline{8} \\ a\,2 \overline{)\ 4\ b} \\ \underline{4} \\ a\,0 \overline{)\ 2} \leftarrow b \end{array} \qquad \begin{array}{r} 2 \\ \\ 2 \\ \\ 2 \\ \\ 2 \end{array}$$

$gcd(24,34) = gcd(10,24) = gcd(4,10)$
$= gcd(2,4) = gcd(0,2)$
$= 2$

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

$gcd(24,34) = gcd(10,24)$
$= gcd(4,10) = gcd(2,4)$

euclidean algo.

//Pseudo code

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

    int temp = gcd(b%a, a)
    return temp;
}
```

T.C: $\log_2 N$ → max(a,b)

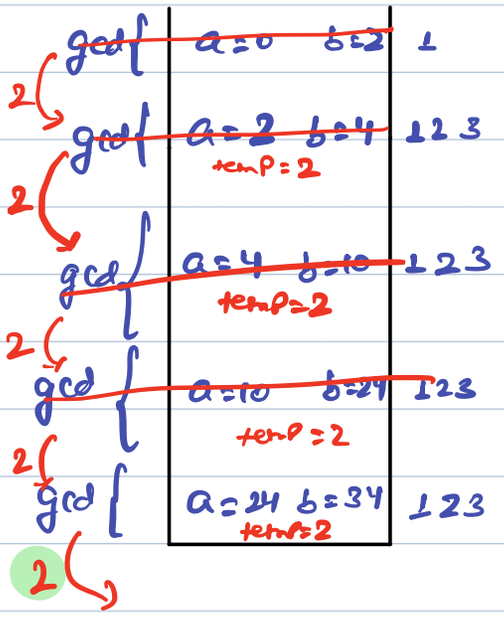
S.C: $\log N$ → stack space

Tracing

$gcd(24,34) = gcd(10,24) = gcd(4,10)$
$= gcd(2,4) = gcd(0,2)$
$= 2$

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

2   int temp = gcd(b%a, a)
3   return temp;
}
```




gcd(b%a, a)

gcd(34,24) = gcd(24,34)
a>b → a<b

Q) Given arr[N], calculate gcd of entire array.

Ex: arr[3] = {6 12 15}
→ 6 → 3

arr[4] = {8 16 12 10}
→ 8 → 4 → 2

// Pseudo Code

```
int gcdarr(int arr[N]){
    int ans = Math.abs(arr[0]);
    for(int i=1; i<N; i++){
        ans = gcd(ans, Math.abs(arr[i]));
    }
    return ans;
}
```

{8 16 12 10}
ans: ~~8~~ ~~8~~ ~~4~~ 2

T.C: O(N log N)
S.C: O(log N) → Stack Space

↳ gcd function written in last problem

→ **Subsequence:** you can select or reject numbers of the array in given order.

arr[4]: { 4 6 3 8 }

→ 4 8 ←
→ 3 ←
→ 8 6 ××
→ 4 6 3 8 ←

arr[2]: {4 6}

— —
— 6
4 —
4 6

arr[3]: {4 6 3}

— — —
— — 3
— 6 —
— 6 3
4 — —
4 — 3
4 6 —
4 6 3

→ No. of subsequence of N length array: $2^N$

Q) Given an array, check if there exists a subsequence with gcd == 1.

Ex: arr[4]: { 4 6 3 8 } → {4,3}, {4,3,8}
↳ true

arr[5]: { 16 10 6 15 27 } → {10,27}
↳ true

arr[4]: { 6 12 3 18 } → false

Idea 1

↳ find all the subsequences and check for gcd == 1.

T.C: $2^n$ * {gcd of subsequences}

XX Idea 2 → if Prime is Present with a different number, answer is true.

↳ The answer can be true even without Prime numbers.

gcd (9, 18) = 9 → odd even combination doesn't work always.

//idea 3

a b c d e f g h

Assume: gcd ( b e h ) = 1

→ gcd ( a b c d e f g h )

→ gcd ( gcd ( b e h ) , gcd ( a c d f g ) )

→ gcd ( 1 , gcd ( a c d f g ) ) = 1

if you have a subsequence with gcd = = 1, gcd of whole array is going to be 1.

//Pseudo code

```
boolean gcdSubs (int arr[N]){
    int ans = math.abs(arr[0]);
    for (int i=1; i<N; i++){
        ans = gcd(ans, math.abs(arr[i]));
    }
    if (ans==1){return true;}
    else {return false;}
}
```

## Q) Delete to maximize

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

Ex:

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 24 | 16 | 18 | 30 | 15 |

* [ ] = 1

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 24 | 16 | 18 | 30 | 15 |

[ ] * [ ] = 3

⇒ ans = 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

//idea1

```
int ans = -∞;
for (int i=0; i<N; i++) {
    //neglect arr[i] & find gcd of rem. elements
    ans = max(ans, x);
}
```

Tc: $O(N * N\log N) = O(N^2 \log N)$

//idea2

| | 0 | 1 | 2 | 3 | 4 ← i |
|---|---|---|---|---|---|
| arr[5]: | 24 | 16 | 18 | 30 | 15 |
| Pgcd[5]: | 24 | 8 | 2 | 2 | 1 |
| Sgcd[5]: | 1 | 1 | 3 | 15 | 15 |

i=0 → Sgcd[1] = 1

i=1 → left = Pgcd[i-1] = Pgcd[0] = 24
right = Sgcd[i+1] = Sgcd[2] = 3

gcd(24, 3) = 3

i=2 → left = Pgcd[i-1] = Pgcd[1] = 1
right = Sgcd[i+1] = Sgcd[3] = 15
gcd(1, 15) = 1

//Psuedo code

```
int delet to maximize (int arr[N]){
    int [] Pgcd = func1 (arr); → H.W
    int [] Sgcd = func2 (arr); → will do
     int ans = 0;
      ans = Math.max (ans, Sgcd[1]);
    for(int i=1; i<N-1; i++ ){
        int left = Pgcd[i-1];
        int right = Sgcd[i+1];
        int overallgcd = gcd (left, right);
          ans = Math.max (ans, overallgcd);
    }
     ans = Math.max (ans, Pgcd[N-2]);

     return ans;
}
```

T.C: O(N log N)

S.C: O(N)

```
int[] func2 (int arr[N]){
    int[] sgcd = new int[N];
    sgcd[N-1]=arr[N-1];
    for(int i=N-2; i>=0; i--){
        sgcd[i] = gcd(sgcd[i+1], arr[i]);
    }
    return sgcd;
}
```

→ leetcode 2447 → [medium]

Q) Number of Subarrays with GCD equals K.

↳ Given arr[N] and K, return the number of subarrays of nums where the gcd equals K.

K=3

arr[6]: { 9 3 1 2 6 3 } → 4

(0 1 2 3 4 5)

→ {3} {3}

{9 3} {6 3}

//idea 1

```
int count=0;
for(int sp=0; sp<n; sp++){
    int ans=0;
    for(int ep=sp; ep<n; ep++){
        ans = gcd(ans, arr[ep]);
        if(ans<k){break;}
        if(ans==k){count++;}
    }
}
```

K=3

arr[6]: { 9 3 1 2 6 3 }

(0 1 2 3 4 5)

sp ↓ ep ↓

ans: (0,3) = (3,1) = 1

count = 0 ✗ 2

T.C: O(N² · log N)

→ easy tag

Q) X of a kind in a deck

↳ you are given integer array deck where deck[i] is a number. Partition the cards into one or more groups. i.e.

- Each group has exactly x card where x>1.
- All cards in one group should have same integers.

ex1: 2 3 8 2 3 8 4 8 4 8 → true

ex2: 2 4 3 2 2 4 5 3 → false

//idea1

2 → 4
3 → 2
5 → 6
10 → 4

| 33 | 22 | 22 |
| 55 | 55 | 55 | 10 10 |
10 10

↳ incorrect idea

→ 2 2 2 2 5 5 5 5 5 5 10 10 10 10

//idea2

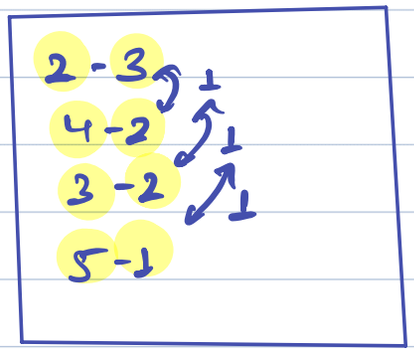
group size = 2 → gcd of occ.
↳ gcd > 1 → true
↳ gcd == 1 → false

2 → 4 ↻2
5 → 6 ↻2 → gs
10 → 4

2 2 | 2 2 | 5 5 | 5 5 | 5 5 | 10 10
10 10

ex1: 2 3 8 2 3 8 4 8 4 8


ex2: 2 4 3 2 2 4 5 3




→ false

// Psuedo code

```
boolean xOfAKind(int arr[N]){
    HashMap<Integer, Integer> map = new HashMap;

    for(int i=0; i<N; i++){
        if(map.containsKey(arr[i]) == false){
            map.Put(arr[i], 1);
        }
        else{
            int temp = map.get(arr[i]);
            map.Put(arr[i], temp+1);
        }
    }

    int ans = 0;
    for(int k : map.keySet())
        ans = gcd(ans, map.get(k));

    if(ans == 1){return false;}
    else {return true;}
}
```

T.C: O(N * log N)

S.C: O(N)

If you didn't come this far only to come this far.