

# Maths1

[Intro To Modular Arithmetic](#)

[Count Pair whose sum%m is 0](#)

[GCD](#)

[GCD\(a, b\)](#)

[Delete one to maximize GCD of all array elements](#)

# Intro To Modular Arithmetic

## Modular Arithmetic

$A \% B$  = Remainder when  $A$  is divided by  $B$ .  
 $[0, B-1]$  range.

why %?

— Helps in limit the range of data.

## Modulus Arithmetic Properties

$$1) (a+b) \% m = (a \% m + b \% m) \% m$$

$$2) (a \times b) \% m = (a \% m \times b \% m) \% m$$

$$3) (a + m) \% m = (a \% m) \quad \text{--- from (1)}$$

$$4) (a - b) \% m = (a \% m - b \% m) \% m$$

Possible it can be negative. So,  
 $= (a \% m - b \% m + m) \% m$ .

$$5) (a \% m) = (a \% m) \% m$$

$$\cancel{6)} (a^b) \% m = ((a \% m)^b) \% m$$

# Count Pair whose sum % m is 0

Problem

Given  $N$  array elements, find count of pairs  $(i, j)$  such that  $(arr[i] + arr[j]) \% m = 0$   $i \neq j$ .

$arr[]: \{4, 3, 6, 3, 8, 12\}$ ,  $m = 6$ .

$4 + 8 \% 6 = 0$   
 $(6 + 12) \% 6 = 0$   
 $(3 + 3) \% 6 = 0$

} 3 pairs.

Brute force

→ Iterate and check each pair

T.C  $\equiv O(N^2)$

S.C  $\equiv O(1)$

Optimised

$(a + b) \% m = 0$   
 → should be multiple of  $m$ .  
 →  $(a \% m + b \% m) \% m = 0$

$arr[] \% m = \{4, 3, 0, 3, 2, 0\} \Rightarrow \{0, m-1\}$

↑ Get this value

$(a + b) \% m = 0$   
 $[0, m-1]$   $[0, m-1]$   
 $[0, 2m-2] \% m = 0$

}  $(a + b) \rightarrow$  can be only 0 or  $m$ .

→ Use hashmap,

index	a	b = m - a
0	4	2
1	3	3
2	0	6
3	3	3
4	1	5
5	0	6

hm  $\{ \}$   
 ans  $\{4:1\}$   
 0  $\{4:1, 3:1\}$   
 0  $\{4:1, 3:1, 0:1\}$   
 0  $\{4:1, 3:2, 0:1\}$   
 1 ← update 1 as I found twice sums = 6.

Code:-

```
hash < int, int > hm;  
for (int i=0; i<N; i++)  
{ int a = arr[i] % m // step 1  
  if (a==0) // step 2  
    b=0  
  else  
    b = m-a  
  if (hm.contains(b) &  
      ans = ans + hm[b];  
  if (hm.contains(a))  
    hm[a]++;  
  else  
    hm.insert({a, 1});  
}
```

T.C:  $O(N)$   
S.C:  $O(N)$

# GCD

GCD (Greatest Common divisor)

HCF (Highest Common factor) same as GCD.

$\text{GCD}(a, b)$  = the greatest divisor / factor that divides both  $a$  &  $b$ .

if  $\text{gcd}(a, b) = x$

$$a \% x = 0$$

$$b \% x = 0$$

$$\text{gcd}(15, 25) = 5$$

$$\begin{array}{r} 1 \quad 5 \\ 3 \quad 25 \\ 5 \quad \\ 15 \end{array}$$

$$\text{gcd}(12, 30) = 6$$

$$\begin{array}{r} 2 \quad 2 \\ 3 \quad 3 \\ 6 \quad 5 \\ 12 \quad 30 \\ 6 \quad \\ 15 \end{array}$$

$$\text{gcd}(10, -25) = 5$$

$$\begin{array}{r} 2 \quad 5 \\ 5 \quad -5 \end{array}$$

$$\text{gcd}(0, 4) = 4$$

$$\begin{array}{r} 0 \quad 1 \\ 1 \quad 2 \\ 2 \quad 4 \\ \infty \end{array}$$

$$0 \% \text{---} = 0$$

$$\text{gcd}(0, -10) = 10$$

$$\begin{array}{r} 2 \quad 10 \\ 1 \quad \\ \infty \end{array}$$

$$\text{gcd}(0, 0) = \infty$$

Properties of GCD

- ①  $\text{gcd}(a, b) = \text{gcd}(b, a)$
- ②  $\text{gcd}(a, b) = \text{gcd}(|a|, |b|)$   $|x|$  = absolute of  $x$
- ③  $\text{gcd}(0, a) = |a|$   $a \neq 0$
- ④  $\text{gcd}(a, b, c) = \text{gcd}(a, \text{gcd}(b, c))$   
 $= \text{gcd}(\text{gcd}(a, b), c)$

# GCD(a, b)



Q write a function to find gcd(a, b)  
Brute force :- for both a, b find all the factors & then find largest common between them.

```
a > 0, b > 0
for (i = min(a, b); i >= 1; i--)
{
    if (a % i == 0 && b % i == 0)
        return i;
}
}
```

Assume  $a > b$ ;  
 $\text{gcd}(a, b) = \text{gcd}(a-b, b)$

①

Using Recursion;

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

Say  $a > b$ ;

$$\begin{aligned}\text{gcd}(a, b) &= \text{gcd}(a-b, b) \\ &= \text{gcd}(b, a-b) \\ &= \text{gcd}(b, a-2b) \\ &= \text{gcd}(b, a-3b) \\ &\vdots\end{aligned}$$

$a - \text{maximal of } b \leq a$

②

$\therefore \text{gcd}(a, b) = \text{gcd}(b, a \% b)$

## Delete one to maximize GCD of all array elements

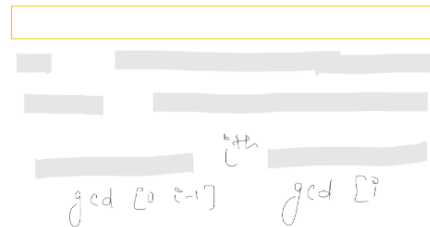
Q Given  $N$  array elements. We have to delete one element such that GCD of remaining element is maximized.

Eg:- arr[]: {24, 16, 18, 30, 15}

After removing 16, GCD will be maximized.

Bruteforce

```
for (i in arr)
  for (j in arr)
    except i, gcd;
  }
ans = max(-
```



Maintain prefix of GCD.  
Maintain suffix of GCD

$P[i]$  = GCD from 0 to  $i$ ;

$P[0] = arr[0]$

```
for (i = 1; i < N; i++) {
  P[i] = gcd(P[i-1], arr[i]);
}
```

$S[i]$  = GCD from  $i$  to  $N-1$

$S[N-1] = arr[N-1]$

```
for (i = N-2; i >= 0; i--) {
  S[i] = gcd(S[i+1], arr[i]);
}
```

```
for (i = 1; i < N-1; i++) {
  x = gcd(P[i-1], S[i+1]);
  ans = max(ans, x);
}
return ans;
```

