Todays Content

1. Sum of xr f all pairs
2.

pow $a^n$ using bit manipulation

1Q Given an ar[N] find sum of xor of all pairs.

**Eg1:**

0 1 2 3

ar[3] = {4 2 5 3} ans = 56

#pairs

| i j: | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | (4^4) | (4^2) | (4^5) | (4^3) |
| 1 | (2^4) | (2^2) | (2^5) | (2^3) |
| 2 | (5^4) | (5^2) | (5^5) | (5^3) |
| 3 | (3^4) | (3^2) | (3^5) | (3^3) |

#Idea1: Create all pairs, for all pairs calculate xor & take sum.

```
#Given ar[N]        TC: O(N^2)  SC: O(1)
int sum = 0;
i = 0; i < N; i++) {
    j = 0; j < N; j++) {
        sum = sum + ar[i] ^ ar[j]
    }
}
return sum;
```

#Idea2:

| i j: | 0 | 1 | 2 | 3 | i | j |
|---|---|---|---|---|---|---|
| 0 | (4^4) | (4^2) | (4^5) | (4^3) | 0 | 1 |
| 1 | (2^4) | (2^2) | (2^5) | (2^3) | 1 | 2 |
| 2 | (5^4) | (5^2) | (5^5) | (5^3) | 2 | 3 |
| 3 | (3^4) | (3^2) | (3^5) | (3^3) | i | i+1 |

```
#Given ar[N]
int sum = 0;        TC: O(N^2/2) = O(N^2)
i = 0; i < N; i++) {
    j = i+1; j < N; j++
        sum = sum + ar[i] ^ ar[j]
    }
}
return 2*sum;
```

**#Idea3:** Contribution = #Add contribution of each element.

#update = #Add contribution of each bit

| | | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| **Ex:** arr[] : { | 15 | 24 | 11 | 19 | 28 | 9 } |

$a \wedge b = 1$

| **#DryRun:** | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|
| | 4 | 3 | 2 | 1 | 0 |
| 15 = | 0 | 1 | 1 | 1 | 1 |
| 24 = | 1 | 1 | 0 | 0 | 0 |
| 11 = | 0 | 1 | 0 | 1 | 1 |
| 19 = | 1 | 0 | 0 | 1 | 1 |
| 28 = | 1 | 1 | 1 | 0 | 0 |
| 9 = | 0 | 1 | 0 | 0 | 1 |

| **#Bits:** | #Set | #UnSet | #Pairs | #Contribution |
|---|---|---|---|---|
| 0 | {15 11 19 9}:4 | {24 28}=2 | 4*2*2 = 16 | $16*2^0$ = 16 |
| 1 | {15 11 19}:3 | {24 28 9}=3 | 3*3*2 = 18 | $18*2^1$ = 36 |
| 2 | {15 28}:2 | {24 11 19 9}=4 | 2*4*2 = 16 | $16*2^2$ = 64 |
| 3 | {15 24 11 28 9}:5 | {19} = 1 | 5*1*2 = 10 | $10*2^3$ = 80 |
| 4 | {24 19 28}:3 | {15 11 19} = 3 | 3*3*2 = 18 | $18*2^4$ = 288 |

**#PseudoCode**

#Add contribution of all bits = 484

```
Iterate on bit positions from [0..31]: # i bit position.
    Iterate on arr[] & calculate no:of elements with $i^{th}$ bit set = c;
    set = c, unset = N - c, pairs = (c)(N-c)*2    # pairs with $i^{th}$ bit = 1
    contribution of $i^{th}$ bit = $2^i * (c)(N-c)*2$
    Add contribution of each bit in ans
}
return ans;
```

```
long pairSum(vector<int> &arr){   TC: O(32N) = O(N)   SC: O(1)
    int N= arr.size();
    long sum=0;
    for(int i=0; i<32; i++){ # i = bit's position.
        # i: Iterate on arr[] & calulate no:of elements with i^th bit set = c;
        int c=0;
        for(int j=0; j<N; j++){
            if( (arr[j]>>i)&1 ==1){
            }    c++;
        }
                          # Pairs with i^th bit set
        sum = sum + c*(N-c)*2 * 2^i
                          # Contribution of i^th bit
    }
    return sum;
}
```

TODO: Sum of and of all pairs : Easy

Sum of OR of all pairs : Medium

2Q Given a, n calculate & return $a^n$

**Ex:** a n

3 5 = $3^5 \rightarrow 243$

2 4 = $2^4 \rightarrow 16$

#**Idea:** Multiply a with itself n times

```
long power(int a, int n){ ✗
    long x = a;
    for(int i = 1; i <= n; i++){
        x = x*x;
    }
}
return x;
```

# a, N=3 ⇒ $x = a^3$

$x = \boxed{\cancel{x}}\ \cancel{a^2}\ \cancel{a^4}\ a^8$

| i | i<=3 | |
|---|---|---|
| 1 | 1<=3 | $x = (x*x) = a^2$ |
| 2 | 2<=3 | $x = (x*x) = a^2 * a^2 = a^4$ |
| 3 | 3<=3 | $x = (x*x) = a^4 * a^4 = a^8$ |

return $x = a^8$

TC: O(N) SC: O(1)

```
long power(int a, int n){ ✓
    long x = 1;
    for(int i = 1; i <= n; i++){
        x = x*a;
    }
}
return x;
```

# #Optizatim:

1. Any +ve N = sum of powers f 2 ✓ ?

   Reason = Every number as binary → sum of powers f 2.

   $10 = 2^1 + 2^3$

   $7 = 2^0 + 2^1 + 2^2$

   $14 = 2^3 + 2^2 + 2^1$

2. $a^{m+n} = a^m * a^n$

   $a^{10} = a^{2^1+2^3} = a^{2^1} * a^{2^3}$

   $a^{13} = a^{2^0+2^2+2^3} = a^{2^0} * a^{2^2} * a^{2^3}$

   $a^{22} = a^{2^4+2^2+2^1} = a^{2^1} * a^{2^2} * a^{2^4}$

$$\boxed{a^{\boxed{N}} = a^{2^0+2^1+\dots 2^n} = a^{2^0} * a^{2^1} * \dots a^{2^n}}$$

**#Con:** If we have $a^{2^0}, a^{2^1}, a^{2^2}$.. using them we can calculate $a^N$

**#Doubt:** which powers to actually use?

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$a^{43}$ ; 1 0 1 0 1 1 $= 2^5 + 2^3 + 2^1 + 2^0$

$p =$ ~~$a^{2^0}$~~ $a^2$ $\qquad n = 1$

| i | Set | $n = n * p$ | $p = p * p$ ; |
|---|---|---|---|
| 0 | ✓ | $n = 1 * a$ ; $a$ | $p = a^1 * a^1 = a^2$ |
| 1 | ✓ | $n = n * a^2$ ; $a^3$ | $p = a^2 * a^2 = a^4$ |
| 2 | ✗ | $n = a^3$ | $p = a^4 * a^4 = a^8$ |
| 3 | ✓ | $n = n * a^8$ ; $a^{11}$ | $p = a^8 * a^8 = a^{16}$ |
| 4 | ✗ | $n = a^{11}$ | $p = a^{16} * a^{16} = a^{32}$ |
| 5 | ✓ | $n = n * a^{32}$ ; $a^{43}$ | $p = a^{32} * a^{32} = a^{64}$ |

return $n$ ; $a^{43}$

| | | | | | |
|---|---|---|---|---|---|
| | 4 | 3 | 2 | 1 | 0 |
| | $a^{16}$ | $a^{8}$ | $a^{4}$ | $a^{2}$ | $a^{1}$ |
| N = 21 $a^{21}$ ; | 1 | 0 | 1 | 0 | 1 |

p = a   ans = 1

| i | set | ans = ans * p | p = p * p; |
|---|---|---|---|
| 0 | ✓ | ✓ ans = a | $p = a \times a = a^2$ |
| 1 | ✗ | ✗ | $p = a^2 * a^2 = a^4$ |
| 2 | ✓ | $ans = a * a^4 ; a^5$ | $p = a^4 * a^4 , a^8$ |
| 3 | ✗ | ✗ | $p = a^8 * a^8 = a^{16}$ |
| 4 | ✓ | $ans = a^5 * a^{16} ; a^{21}$ | $p = a^{16} * a^{16} = a^{32}$ |

return $ans = a^{21}$

```
# Given a, N;
long p = a, ans = 1;
for(int i=0; i<32; i++){
    if((N>>i)&1 == 1){
        ans = ans * p;
    }
    p = p * p;
}
return ans;
```

```
# Given a, N;
long p = a, ans = 1;
for(int i=0; i<32; i++){
    if((N>>i)&1 == 1){
        ans = ans * p;
    }
    p = p * p;
}
return ans;
```

```
# Given a, N; Binary Exponentiation
long p = a, ans = 1;
while(N>0){    TC: O(log_2 N)  SC: O(1)
    if(N&1 == 1){
        ans = ans * p;
    }
    p = p * p;
    N = N>>1; # N = N/2
}
return ans;
```

2Q Given ar[N] it contains all elements from 1.. N.

#1 element from 1 to N repeats

#1 element from 1 to N missing

Return both repeat & missing element

Note: No Extra space, No modifying array.

**Constraints:**

$1 \le N \le 10^6$

$1 \le ar[i] \le N$.

**Ex:** missing repeat

ar[5] = { 2 2 1 4 5 }

ar[7] = { 1 3 6 5 4 6 7 }

**Idea:**