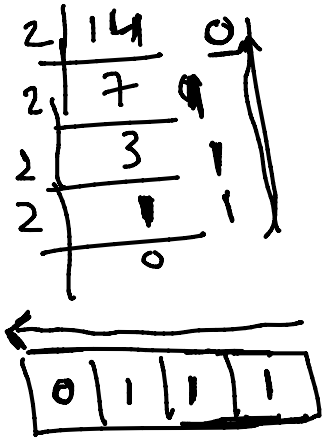
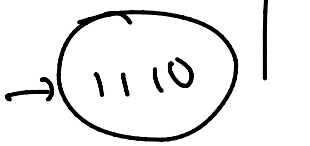
CLASS 43

27 March 2023 21:08

Class-43






```
vector<int> bits;
int n;
while (n != 0)
{
    int rem = n%2;
    bits.push_back(rem);
    n = n/2;
}
reverse(bits.begin(), bits.end());
```

14/2 = (7)

7/2 = 3

3/2 = (1)

1/2 = (0)

18 →

1 1 0 0 0 1 0 0 0

```
for(i=0; i<=31; i++)
{
    if((n>>i)&1)
    {
        cout<<"set bit"; break;
    }
}
```

Toggle bits b/w given Range

N = 17

L = (2) R = (3)

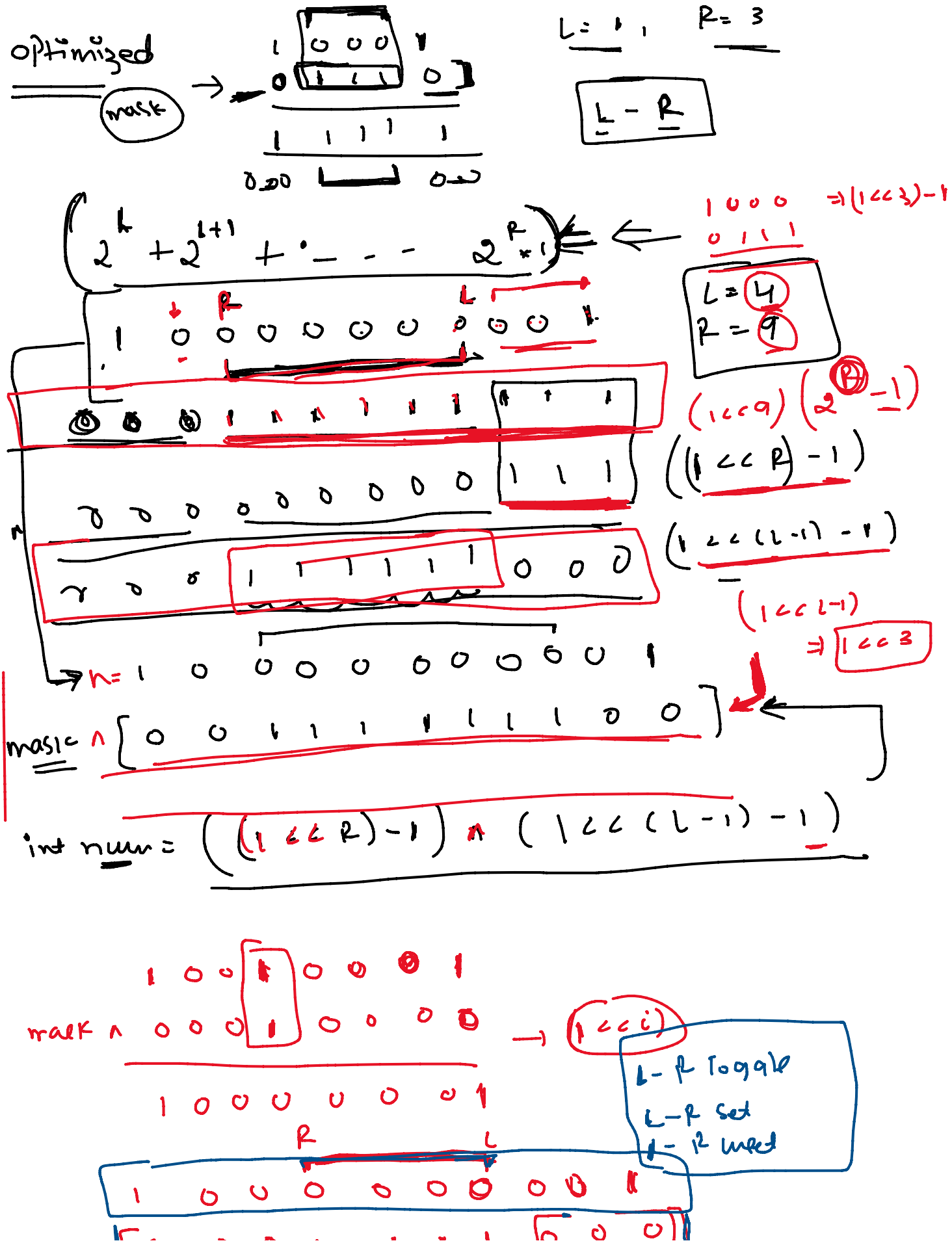
```
l--; r--;
```

1 0 0 0 1 → 1 0 1 1 1

```
for(i=l; i<=r; i++)
{
    n = n ^ (1<<i)
}
```

XOR: 1 ^ 1 = 0, 0 ^ 1 = 1

Toggle


optimized
mask
L = 1, R = 3
L - R
1000 ⇒ (1<<3)-1
0111
L = 4
R = 9
(1<<9) (2^9 - 1)
((1 << R) - 1)
(1 << (L-1) - 1)
(1<<L-1)
⇒ 1<<3
mask ^
int num = (((1 << R) - 1) ^ (1 << (L - 1) - 1))
mask ^
(1<<i)
L-R toggle
L-R set
L-R unset

To create the mask → 0 - R [all 1s]

0 - L-1 [all 1s]

FirstNO → (1 << R) - 1

SecondNO → (1 << (L-1)) - 1

mask = FirstNO ^ SecondNO

n = n ^ mask

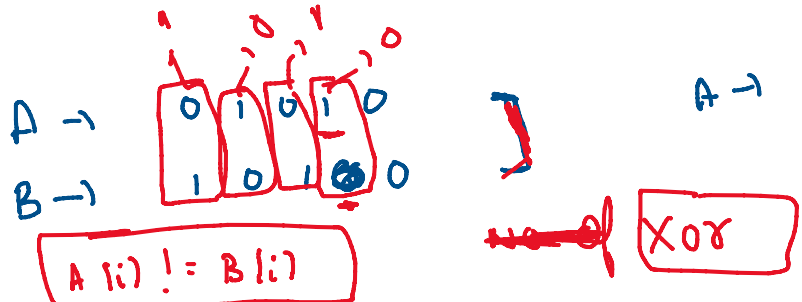
Toggle, Set, Unset → n & (1 << i)

(1 << i) ^ n

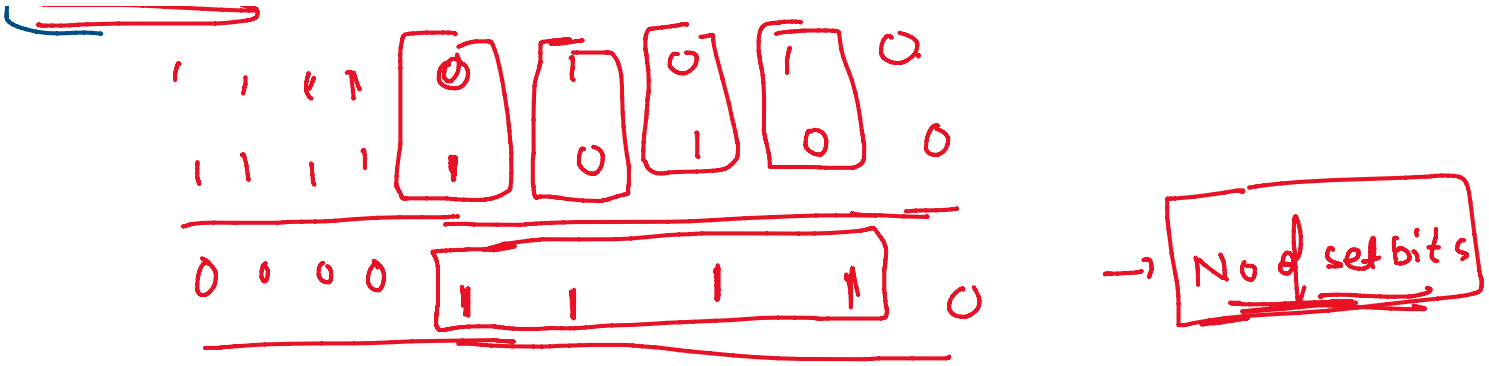
n | (1 << i)

OA →

For ( )



approach-1 A ^ B



31

Approach 2

A → 1 1 1 1 0 1 0 1 0

B → 1 1 1 1 1 0 1 0 1

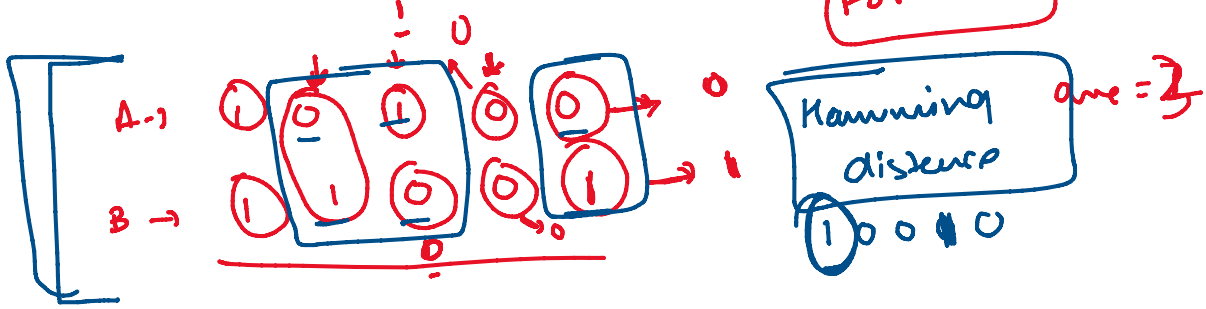
How to check if ith bit is set or Not

$(n >> i) \& 1$ or $n \& (1 << i)$

```
For ( i = 0 ; i <= 30 ; i++ )
{   if ( (a >> i) & 1  ==  (b >> i) & 1 )
    {   continue ;
    }
    else
        ans++ ;
}
```

if ith bit is set

a



[4, 14, 2] → (4,2) + (4, 14) + (14, 2)

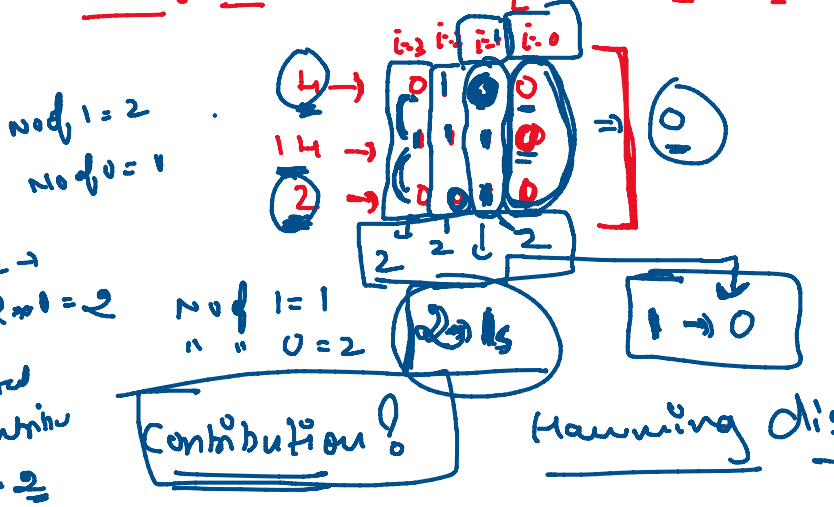
Brute Force

```
For (i = 0; i < n; i++)
    For (j = i+1; j < n; j++)
    {
        ans += Calculate_Hamming_Distance(nums[i], nums[j]);
    }
```

$O(n^2 * \log n)$

$O(n^2 \log n)$

Optimized ? → [4, 14, 2]



No of 1 = 2
No of 0 = 1

i=2 → 2 & 1 = 2

No of 1 = 1
" " 0 = 2

Total contribution = 2

Contribution ?

(4, 14) = 0th bit ⇒ 0
(4, 2) = 1
(14, 2) = 1

2 & 1 = 0

Hamming distance

Total contribution for i=1 & i=2

Total_contribution for i^{th} bit =

No of 1s * No of 0s

i^{th} bit → 1
→ 0

Code

```
vector<int> v(32, 0);
for (int i = 0; i < 30; i++)
{
    for (int j = 0; j < nums.size(); j++)
    {
        if (nums[j] & (1 << i))
        {
            v[i]++;
        }
    }
}
```

$30 * O(n)$

$O(30n)$

30-

O(30*n)

n = 10^4

```
        v[i]++;
    }
}
int ans = 0
for (i=0; i<31; i++)
{ ans += v[i] * (num.size() - v[i]);
}
```

10 → (4)

(10 - 4) → 6

XOR Properties

$x \wedge 0 = x$

$x \wedge x = 0$

```
  1
^ 1
---
  0
```

```
  1
  0
---
  1
```

→

```
  1 1 1 0 1 0 1 0
^ 0 0 0 0 0 0 0 0
---------------
  1 1 1 0 1 0 1 0
```

```
  1 1 1 0 1 0 1 0
^ 1 1 1 0 1 0 1 0
---------------
  0 0 0 0 0 0 0 0
```

Ques Given an Array with duplicates except 1 element find that element

→ [(2) 2 (3) 3 (4) (4) 5]

```
int ans = a[i];
For (i=0; i<n; i++)
{ ans ^= a[i];
}
```

ans = 0 ^ 3 = 3

ans = 0

0 ^ 4 = (4) ^ 4 = 0 ^ 5 = (5)

4

$6 \wedge 4 = 4 \quad 4 \wedge 4 = 0 \wedge 6 = 6$

arr: 2 ∧ 2 ∧ 5 ∧ 3 ∧ 4 ∧ 4 ∧ 6

2 2 5 3 3 4 4

arr: 0 ∧ 5 ∧ 3 ∧ 4 ⇒ 5

6

2 5 2 3 4 3 4

ind arr = 2

$a \wedge b = b \wedge a$

$a \wedge b \wedge c = a \wedge c \wedge b$

arr: 4 ∧ 5 ∧ 2 ∧ 3 ∧ 4 ∧ 3 ∧ 4

ans: 5

1 2 3 3 4

⇒ Check Power of 2

⇒ 16 ⇒ 2^4

17* ⇒ 2^4

n & (n-1)

n → 16 → 10000

n-1 → 15 → 01111

00000

n → 17

→ 10001

10000

10000 → 16

2^4

No of setbit = 1

n - 18 → 10010

No of set bits = ?

n = 18 → 10010

n-1 = 17 → 10001

10000 → 16

16 = 8 = 4 = 2 = 1

16

```
while (n & 2 == 0)
    n = n >> 1;
n
if (n == 1)
```

log 2

ith bit

0th bit ⇒ 0 or 1

n & 1

```
while ((n & 1) == 0)
    n >>= 1
```

0 0 0

Swap 2 nos w/o using 3rd variable

a = 10
b = 5

```
temp = a
a = b
b = temp
```

a, b

a = a ∧ b

b = b ∧ a ⇒ b ∧ a ∧ b = a

a = a ∧ b ⇒ a ∧ b ∧ a

a = 10 → 01010
b = 5 → 00101

a = 4, b = 6

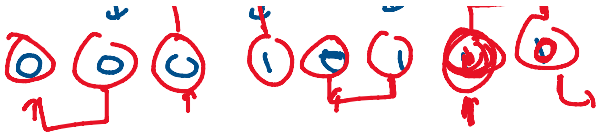
```
a = a+b = 10
b = a-b = 10-6 = 4
a = a-b = 10-4 = 6
```

a = a+b = 4+6

b = a/b = (4+6)/6 = 4

a = (4+6)/4 = 6

(i-1) → unset bit



(i-1) → ~~unset~~ set
(i) → unset

(i), (i-1)

unset set

Fo

[2, 7, 14, 8, 9]

(i - 1) Prefix in (i) ∧ Prefix in (i-1)