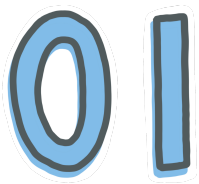
Bit Manipulation 2



Agenda

- Check bit / Count set bits
- Set i^{th} bit
- Unset i^{th} bit
- Set continuous x & y bits

Quick Revision

Quiz 1:

a = 15

print(a<<2)

$a << b = a \times 2^b$

$15 << 2 = 15 \times 2^2$

$= 15 \times 4$

$= 60$

Quiz 2:

int a = 29

print(a>>2)

$a >> b = \frac{a}{2^b}$

$29 >> 2 = \frac{29}{2^2} = \frac{29}{4}$

$= 7$

Quiz 3:

Which of the following options output is 2 power n

$a << b = a \times 2^b$

$1 << N = 1 \times 2^N = 2^N$

$2^{30} = 1 \times 2^{30} = 1 << 30$

$2^{31} = 1 \times 2^{31} = 1 << 31$

Quiz 4:

$a << b = a \times 2^b$

Which of the following options value is 5 power n → 5^N

✗ $5 * (1 << N) = 5 \times (2^N)$

✗ $5 << N = 5 \times 2^N$

✗ $5 << (N-1) = 5 \times 2^{N-1}$

✓ None of the above

Q1 Given N and i, check if i^{th} bit position is set or not.

Set bit = 1 → Unset Bit = 0

Example

N = 21

i = 2

Set

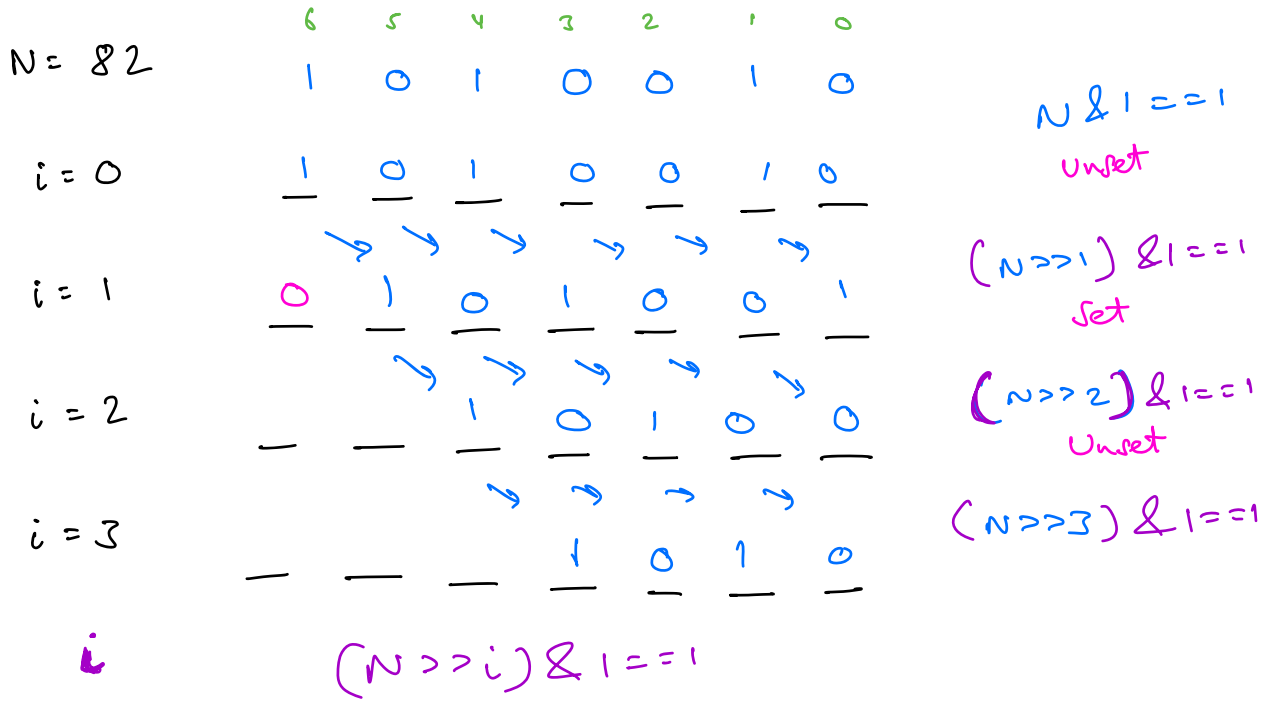
Example

N = 34

i = 3

Unset

Idea



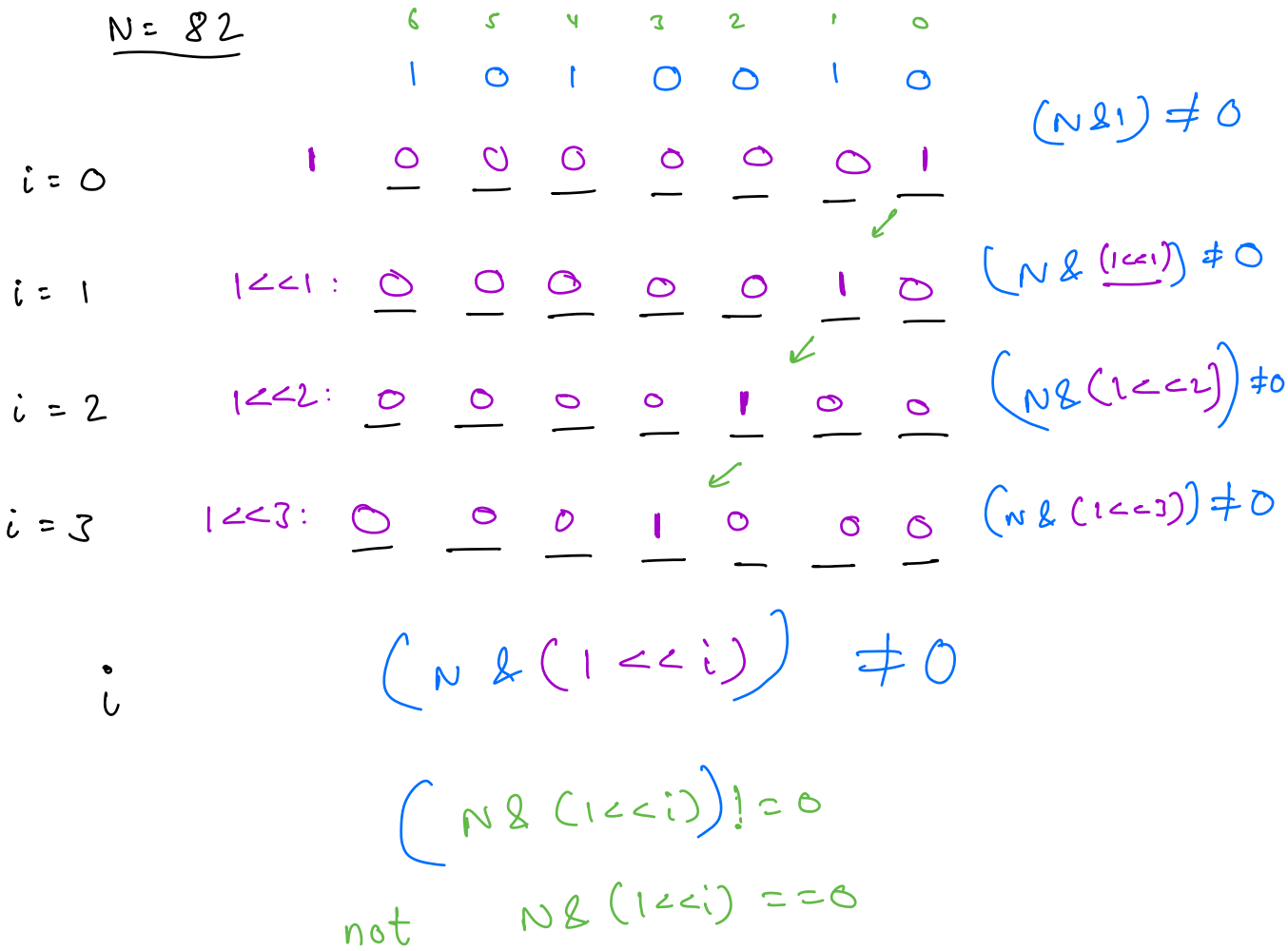

```
Boolean checkBit(int N, int i) {
    if ( (N >> i) & 1 == 1)
        return True
    else
        return false
}
```

return (N >> i) & 1 == 1

TC: O(1)

SC: O(1)

Can we also do it with left shift ?



```
Boolean checkBit(int N, int i) {
        return  ( N & (1<<i) ) != 0
}
```

TC : O(1)

SC : O(1)

Q2 Number of 1 Bits

Qualcomm

Facebook box

Given an integer N, count how many set bits are there in N (Assume N to be a 32 bit integer)

Example

N = 10

4 bits

⇒ 2

Example

N = 27

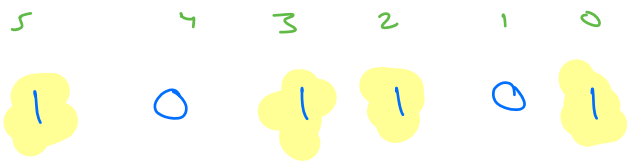
5 bits

⇒ 4

Example

N = 45

6 bits



⇒ 4

n bits → (n-1)

Highest bit

Idea 1

Iterate & check for every bit if it is set or not

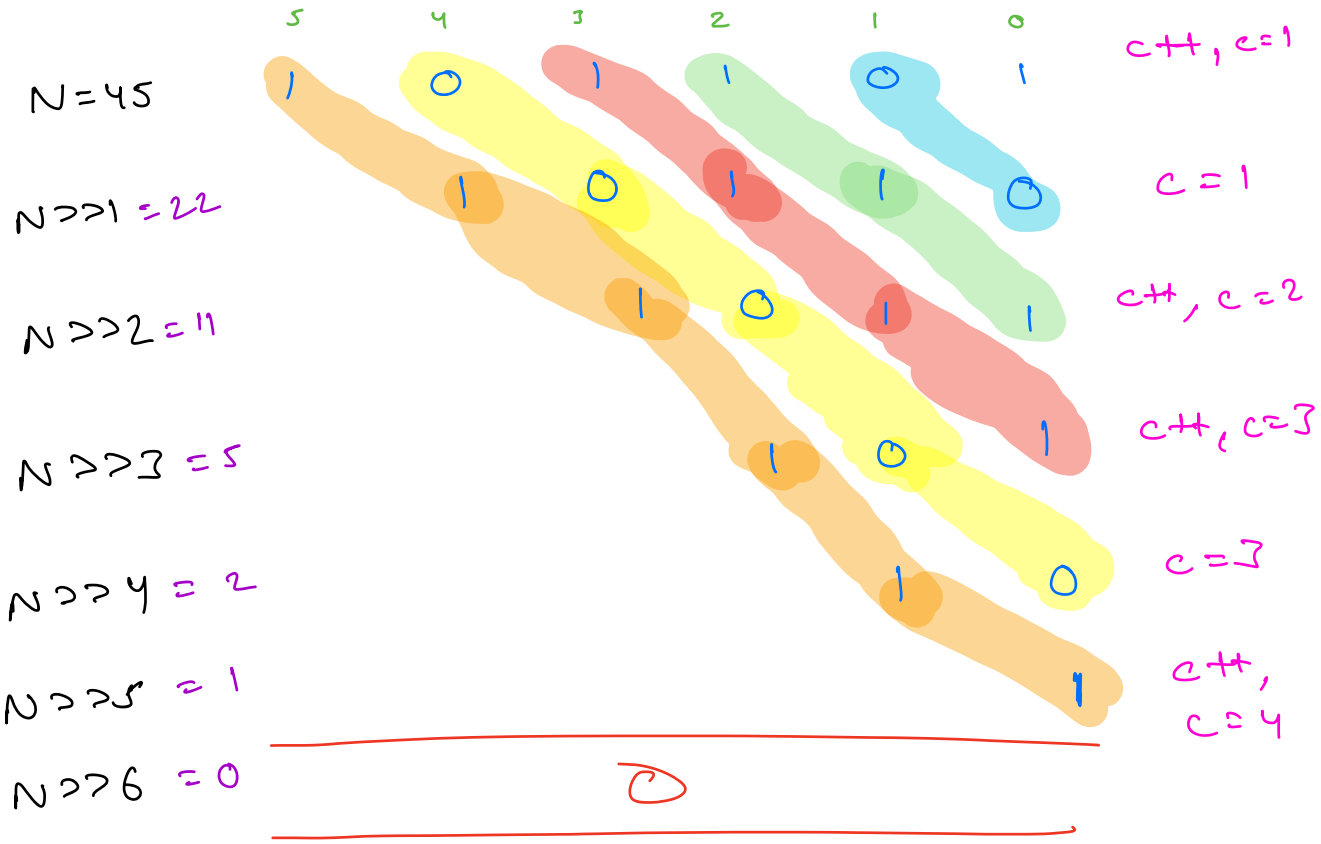
32 bit Number

```
int countSetBits(int N) {
    int c = 0
    for( i=0 ; i<32; i+=1 ) {  -> [0 to 31]
        if ( checkBit(N, i) )
            c++
    }
    return c
}
```

TC: O(1)

32 iterations

Idea 2




```
int countSetBits(int N) {
    int c=0
    while ( N > 0 ) {
        if((N & 1) == 1 )
            c++
        N = N>>1
    }
    return c
}
```

N = N/2

N = 45 → 12 → 22 → 12 → 11 → 12 → 5 → 12 → 2 → 12 → 1

TC: $O(\log_2 N)$

Which approach is better ?

	Idea 1	Idea 2
	$O(1)$	$O(\log_2 N)$
$N = 10$	32 iterations	4 iterations
$N = 30000$	32 iterations	15-16 iterations
$N = 2^{32}$	32 iterations	32 iterations

Exceptional case - Big O gives us the wrong idea.

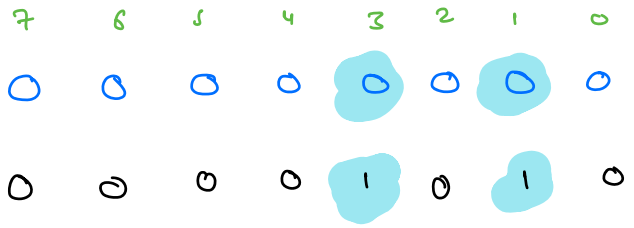
Break till 10 : 25 PM

Q3 Given x & y, set x^{th} bit and y^{th} bit in value 0.

Example

x = 3

y = 1



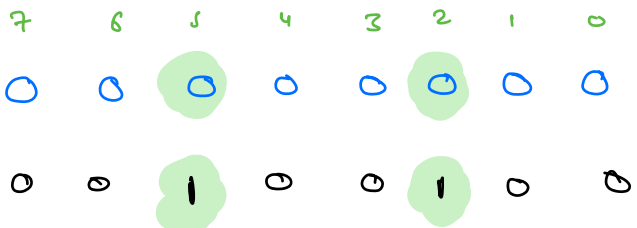
$= 2^3 + 2^1$

$= 10$

Example

x = 5

y = 2



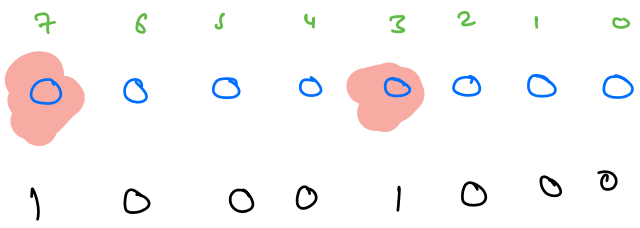
$= 2^5 + 2^2$

$= 32 + 4 = 36$

Example

x = 3

y = 7



$= 2^7 + 2^3$

$= 136$

$ans = 2^x + 2^y$

ans = (1<<x) + (1<<y)

What if x == y ?


```
if (x == y)
    return (1 << x)
else
    return (1 << x) + (1 << y)
```

This works for all cases

No if - else

Madhur's Solution

```
int setBits (int x, int y) {
    return (1 << x) | (1 << y)
}
```

Bitwise OR

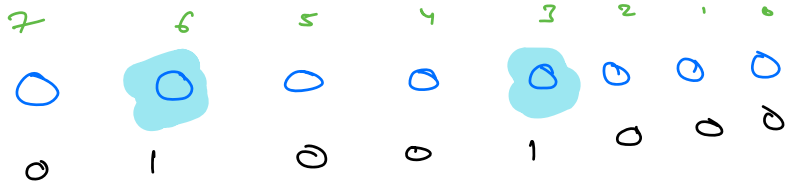
TC: O(1)

SC: O(1)

Example

$x = 6$

$y = 3$



↳ $2^6 + 2^3$

$2^6 = 1\ 0\ 0\ 0\ 0\ 0\ 0$

OR $2^3 = 0\ 0\ 0\ 1\ 0\ 0\ 0$

$1\ 0\ 0\ 1\ 0\ 0\ 0$

Example

$x = 3$

$y = 3$

Expected result

$= (1 << x) = 2^x$

$= 2^y$

If either is 1,
ans = 1
else
ans = 0

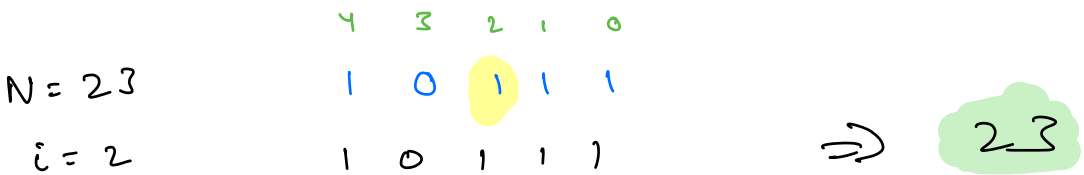
$2^x | 2^y$ $\quad 2^x\ 2^3 = 1 << 3 = 1\ 0\ 0\ 0$

$2^3\ 2^3 = 1 << 3 = 1\ 0\ 0\ 0$

$2^x = 2^y = \leftarrow \quad 1\ 0\ 0\ 0$

$a | a = a$

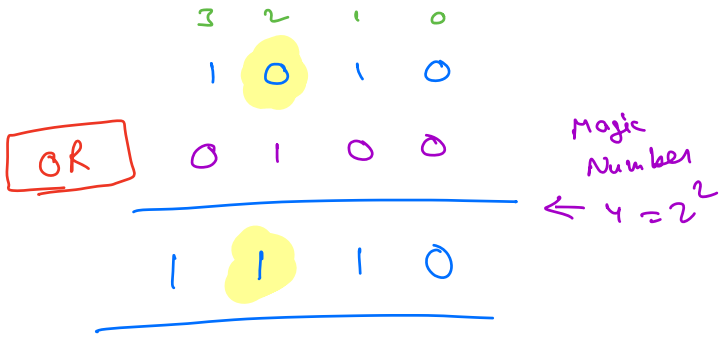
Q4 Given N & i, set the i^{th} bit in N.




```
int setIthBit(int N, int i) {
    return (N | (1<<i))
}
```

N = 10

i = 2



AND

OR

XOR

~~Invert~~

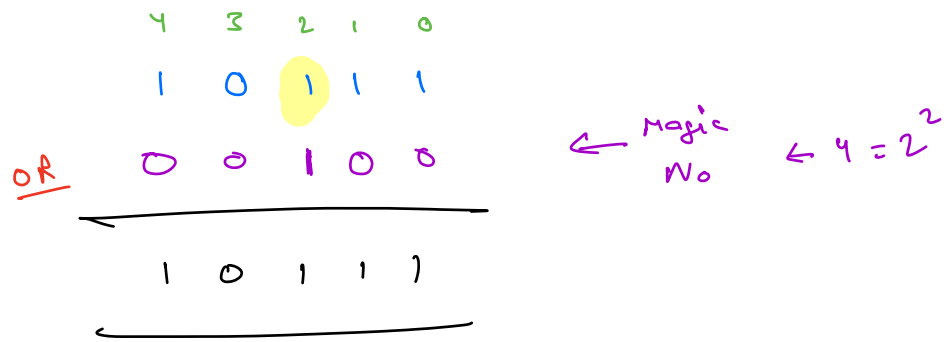
~~<<~~

~~>>~~

Magic Number - Only i^{th} bit set
Rest all bits were unset

N = 23

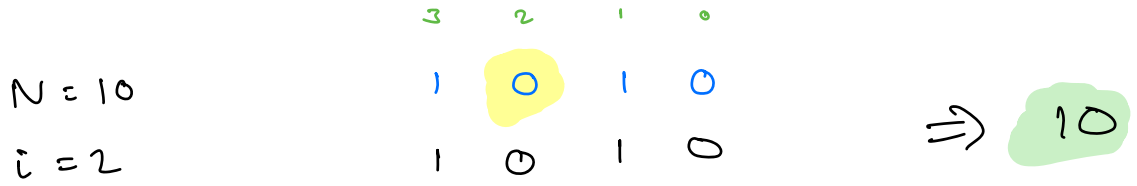
i = 2

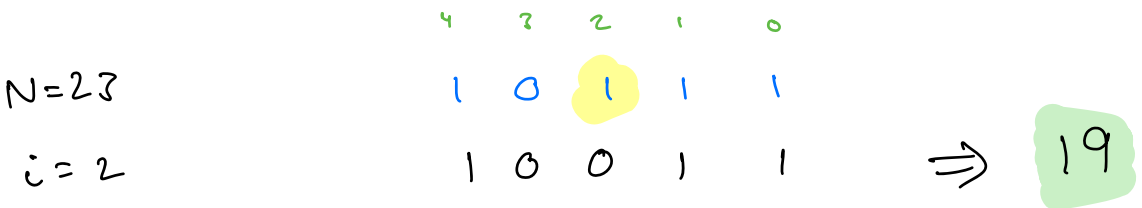


i = 5 → 1 0 0 0 0 0 → 2^5

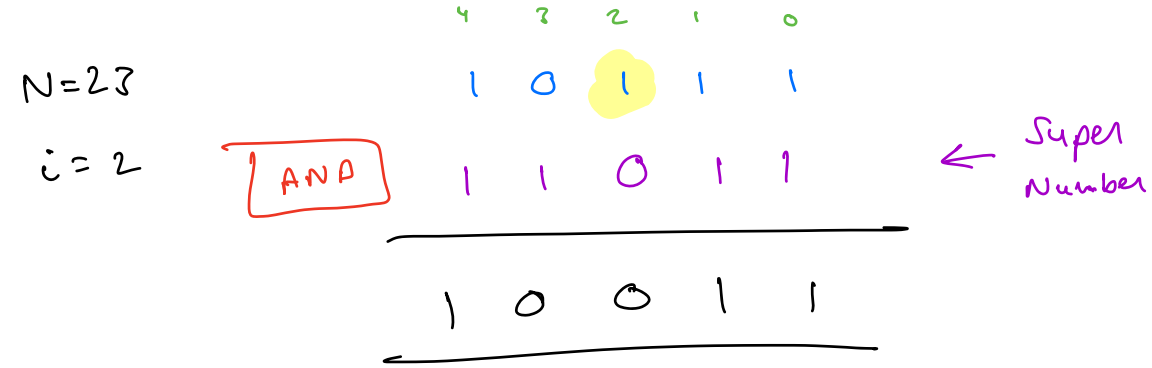
Magic No = 2^i = 1<<i

Q5 Given N & i, unset the i^{th} bit.



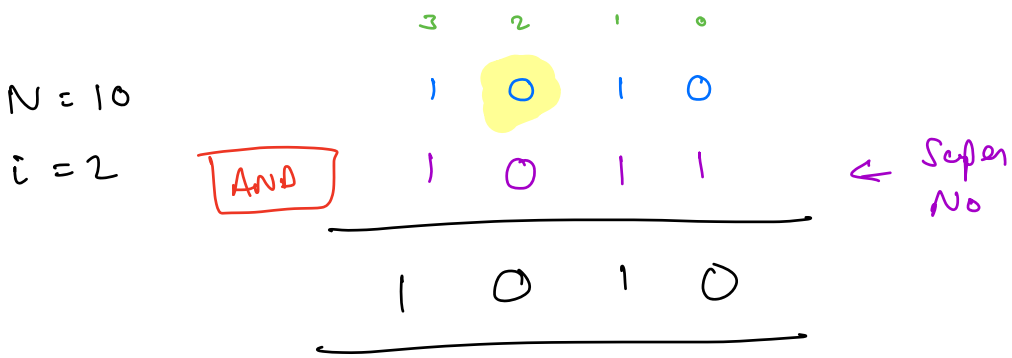



```
int unsetIthBit(int N, int i) {
    return  N & (~ (1<<i ))
}
```



Super No = Only i^{th} bit unset. Rest all bits are set.

AND
OR
XOR
~~NOT~~
~~<<~~
~~>>~~



Result = N & (Super No.)

Super No = Only i^{th} bit unset. Rest all bits are set.

1011

Magic Number - Only i^{th} bit set, Rest all bits were unset $= 1 << i$

$i = 2$

00100

Super No = $\sim$ (Magic No)

$= \sim (1 << i)$

Todo

Q1 Given x & y, set x continuous bits & unset y continuous bits

x = 3
y = 2

1 1 1 0 0 ⇒ 28

x = 4
y = 2

1 1 1 1 0 0 ⇒ 60

x = 1
y = 3

1 0 0 0 ⇒ 8

Easy Level - Solve it however you can

Medium Level - Max 2 lines of code

Q2 Toggle i^{th} bit

→ If i^{th} bit is set, unset it
→ If i^{th} bit is unset, set it

One line code
No if else

Doubts

Good Night

Thank You

Wednesday