Today's Content

→ >> Right shift

→ checkbit / Count Set bits

→ Set $x^{th}$ and $y^{th}$ bit

→ Unset $i^{th}$ bit

→ Set Continuous x and y bits

→ MSB

f

$a << 2 \rightarrow a * 2^2$

$a << n \rightarrow a * 2^n$

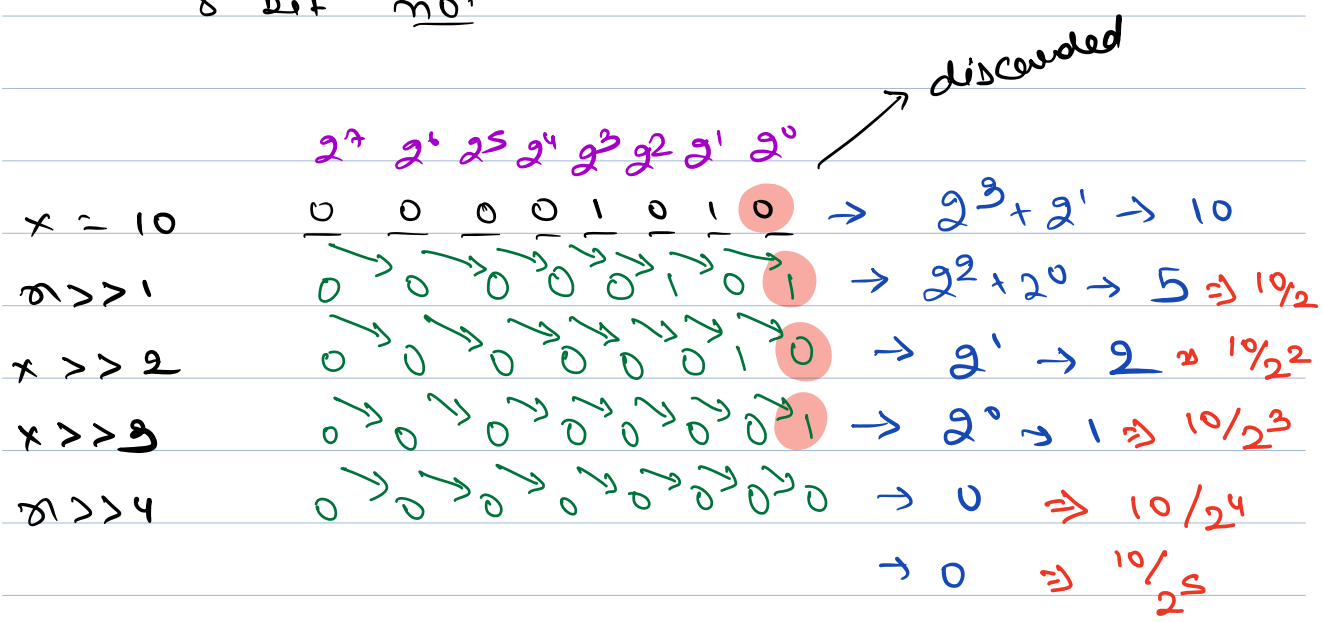
Let's say, $a = 1$,

$1 << 2 \rightarrow 2^2$

$1 << n \rightarrow 2^n$

$2^4 \rightarrow 1 << 4$

8 bit no:




// generalize

$x >> 1 \Rightarrow \frac{x}{2}$ ,

$x >> 2 \Rightarrow \frac{x}{2^2}$

$x >> i \Rightarrow \frac{x}{2^i}$

Bit positions.

int a → 4 Bytes → 32 bits : [31 - 0]

long a → 8 Bytes → 64 bits : [63 - 0]

Ques) Given N, i, check if ith bit pos in N is set or unset.

set → 1, unset → 0

| | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|
| N = 21 : | 1 | 0 | 1 | 0 | 1 |

i = 2 True.

| | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| N = 34 : | 1 | 0 | 0 | 0 | 1 | 0 |

i = 3 false

Brute force idea :-

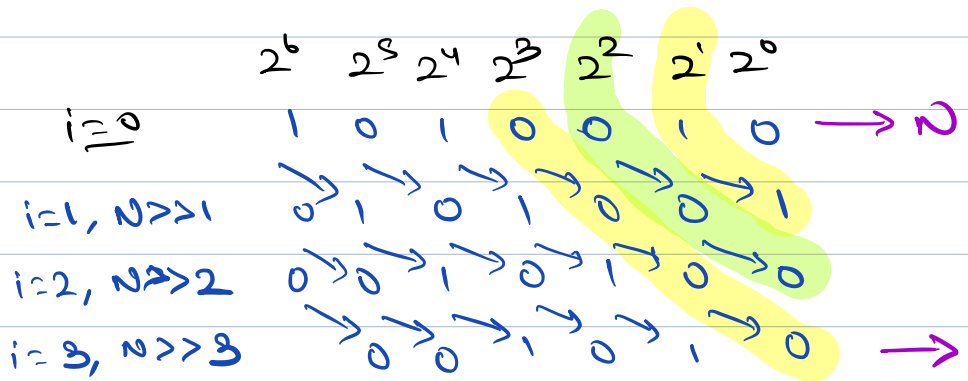
idea :- Convert decimal → Binary and store it & check if its ith bit is 1 or not.

(Last bit 9 knows how to check)

if $0^{th}$ bit of a is 0
then, $(a \& 1) == 0$
→ true

otherwise we get false.

N = 82, i th




i, N>>i {$i^{th}$ bit in N is now $0^{th}$ bit}

```
bool checkBit(int N, int i) {
    if ((N>>i) & 1) == 0) {
        return false;
    } else { return true; }
}
```

12

1

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

T.C → O(1)

S.C → O(1).

(N&1) == 1,

set

N = 82

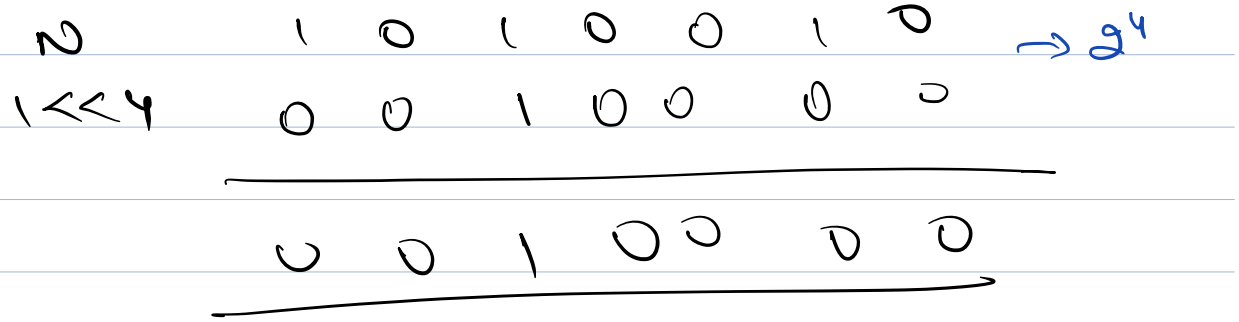
| | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | |
|---|---|---|---|---|---|---|---|---|
| N = | 1 | 0 | 1 | 0 | 0 | 1 | 0 | |
| i = 0, | 0 | 0 | 0 | 0 | 0 | 0 | 1 | ← 1 |
| i = 1, | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 1<<1. | 0 | 0 | 0 | 0 | 1 | 0 | 0 | |

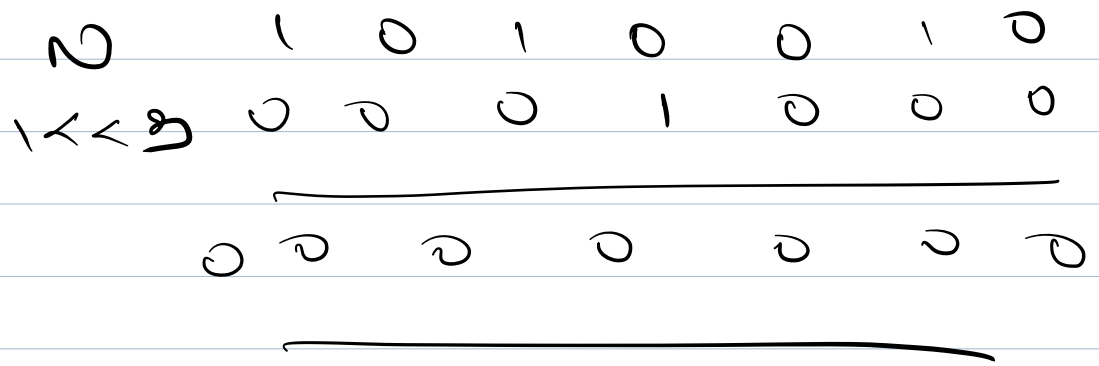
i = 2

1<<2

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| N => | 1 | 0 | 1 | 0 | 0 | 1 | 0 | |
| 1<<1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 1 | 0 | → $2^1$ |

6 5 4 3 2 1 0



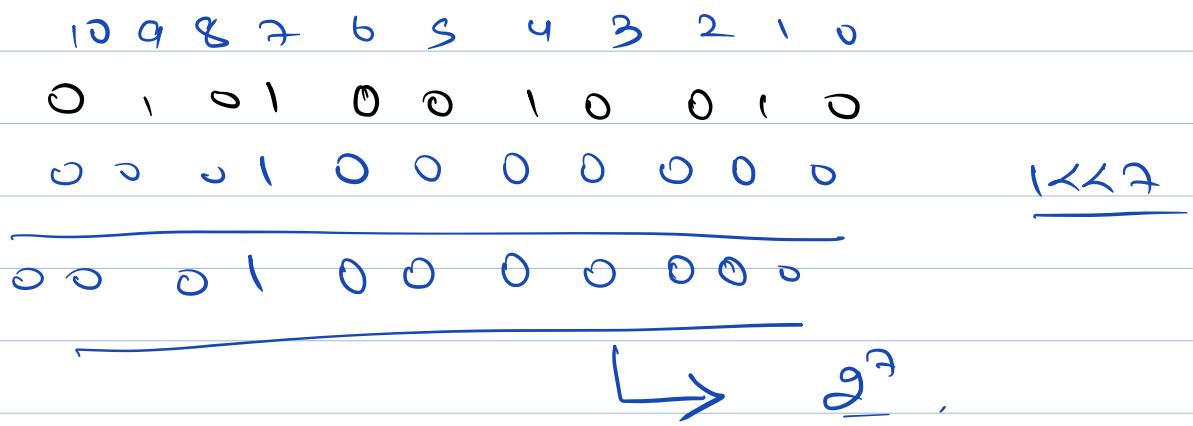




// check bit using left shift.

```
bool checkbit (n, i) {
    if (n & (1<<i) == 0) {
        return false
    } else {
        return true
    }
}
```

T.C → O(1)
S.C → O(1)

return !n & (1<<i) == 0




```
if ((N & (1<<i)) == (1<<i)) {
    return true
}
else { return false }
```

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

10:05 → 10:15 break

$2^5$ → pow(2,5)

1<<5

## Count Set Bits :-

→ Given N integer, calculate how many set bits in N.

| | | Output |
|---|---|---|
| N = 10 | → 1 0 1 0 | 2 |
| N = 27 | → 1 1 0 1 1 | 4 |

Approach 1 :-

Convert to binary and count 1's.

Approach 2 :-

32 bits : [0 - 31]

```
int c = 0
(i = 0; i < 32; i = i+1)
        if (checkBit(N, i))
            c = c+1
return c.
```

T.C → O(1)

S.C → O(1)

## Approach 3 :-

N = 45. C = 0

| 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 | C++, C = 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | |
| 0 | 0 | 1 | 0 | 1 | 1 | C++, C = 2 |
| 0 | 0 | 0 | 1 | 0 | 1 | C++, C = 3 |
| 0 | 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 1 | C++, C = 4 |
| 0 | 0 | 0 | 0 | 0 | 0 | |

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

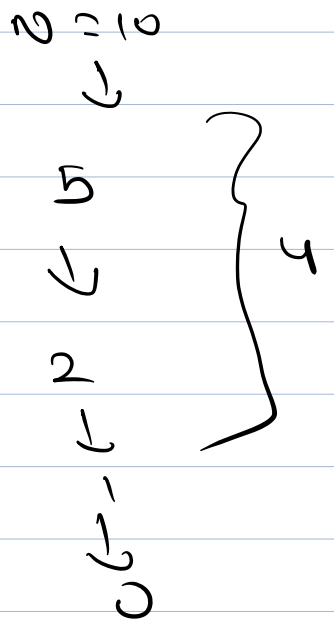
T.C → $(\log_2 N)$

S.C → O(1).

n = 10

↓

32 iterations.




Ques) Given n, y, set $n^{th}$ bit and $y^{th}$ bit, initial value is 0.

n = 3

y = 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

↓

decimal → 10.

$2^3 + 2^1$

$x = 5$

$y = 2$

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

$2^5 + 2^2 \Rightarrow 36$.

$x = 7$

$y = 3$

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

$\rightarrow 2^7 + 2^3 \Rightarrow 136$.

$x = 2$

$y = 6$

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

$\rightarrow 2^1 + 2^2 \Rightarrow 68$

```
int setbits(int x, int y){
    // return 2^x + 2^y
    return (1<<x + 1<<y)
}
```

3

x = 3
y = 3

Expected → 8
Code → 10

```
int SetBits(int x, int y){
    if(x==y) { return 1<<x; }
    else // { return 2^x + 2^y
        return (1<<x + 1<<y)
    }
}
```

x = 6
y = 3

1<<6
1<<3

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

x = 4
y = 2

1<<4
1<<2

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

x = 2   1<<2   0 0 0 0 0 1 0 0

y = 2   1<<2   0 0 0 0 0 1 0 0

0 0 0 0 0 1 0 0

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

3

// given N, set ith bit.

N = 10,   →   3 2 1 0 : 1 0 1 0

i = 2

1 1 1 0 ← ans

N = 23   4 3 2 1 0 : 1 0 1 1 1

i = 2

10 1 1 1

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

```
        4 3 2 1 0
N = 21 → 1 0 1 0 1
i = 3  → 0 1 0 0 0   (1<<3)
         ---------
         1 1 1 0 1
         ---------
```

**Todo :-** given N & i, unset ith bit in N.

```
N = 29      4 3 2 1 0
            1 0 1 1 1
i = 2,        ↓
            0
          -------
            1 0 0 1 1
          -------
```

**Ques** Given x, y set x continuous bits @ y unset bit.

x = 3, y = 2 :- 4 3 2 1 0 → 1 1 1 0 0 → 28

$x = 4, y = 2$ :- 1 1 1 1 0 0 → 60

$x = 2, y = 4$ → 1 1 0 0 0 0 → 48

Before next class → revise everything.

↓

Bits.

1 0 1 1 1 → str1

1 0 1 1 0 → str2

1 1 → 0 → carry = 1

1 0 → 1 → carry = 0

0 0 → 0 → carry = 0

for ( i = n-1 ; i >= 0 ; i-- ){

ch = str1.

ch = 48.

sum = ch1 + ch2 + carry

ans = 0

carry = 1

Decimal = 10 → 32 bits

int

1010

↓

$10 * 2^0$

→ 32

$2^{32}$

→ 1010.

→ 4