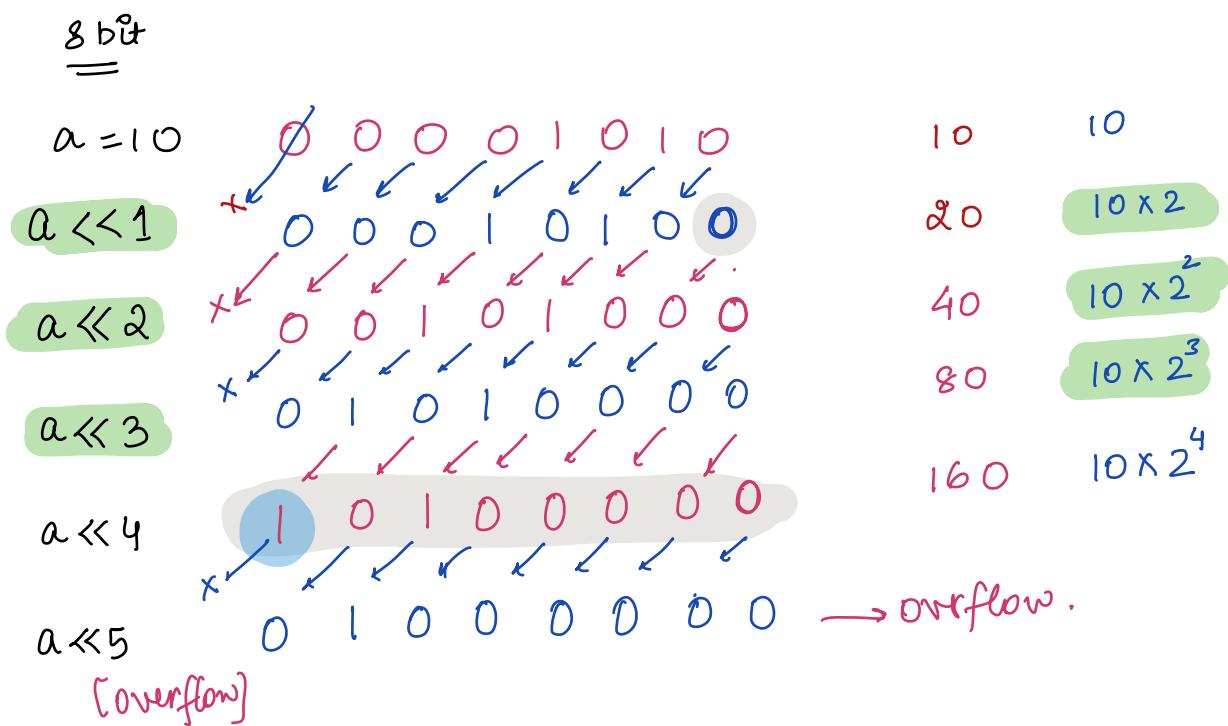


left shift operator
 right shift operator
 Bit Masking
 problems

Left Shift Operator. \ll
 literally shifts bits towards left.



In general, $\| a \ll n = a * 2^n \|$

$$1 \ll n = \underline{\underline{1 * 2^n}}$$

$a = 1$

$$a \ll 1 = 1 * 2^1$$

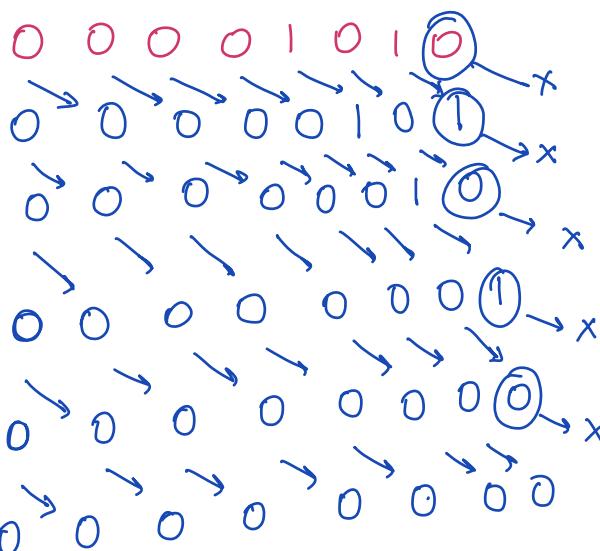
$$a \ll 2 = 1 * 2^2$$

$$a \ll 3 = 1 * 2^3$$

$$2^n = 1 \ll n$$

Right Shift

$a = 10$



$a \gg 1$

$a \gg 2$

$a \gg 3$

$a \gg 4$

$a \gg 5$

10 [$10/2^0$]

5 [$10/2^1$]

2 [$10/2^2$]

1 [$10/2^3$]

0 [$10/2^4$]

0 [$10/2^5$]

$$\| a \gg n = a/2^n \|$$

QUIZ 1

$$a \ll n = a * 2^n$$

$$15 \ll 2$$

$$15 * 2^2 = 60$$

QUIZ 2

$$a \gg n = a / 2^n$$

$$29 >> 2$$

$$29 / 2^2 = \frac{29}{4} = 7$$

$$\underline{a \ll n = a * 2^n}$$

$$1 \ll n$$

$$5 \ll n \rightarrow \underline{5 * 2^n}$$

QUIZ $5^n = ?$

$$\left\{ \begin{array}{l} 5 * (1 \ll n) \longrightarrow 5 * (2^n) \times \\ 5 \ll n \longrightarrow 5 * (2^n) \times \\ 5 \ll (n-1) \longrightarrow 5 * 2^{n-1} \times \end{array} \right.$$

Amazon

Question Given vet nos. N & i, check if ith bit is set or not in N.

$$N = 21 \quad i = 2$$

$$21: \quad \begin{matrix} 4 & 3 & 2 & 1 & 0 & 1 \end{matrix}$$

↓
set / true.

$$N = 25 \quad i = 1$$

$$25: \quad \begin{matrix} 4 & 3 & 2 & 1 & 0 & 1 \end{matrix}$$

↓
unset / false

Have we done something to check the rightmost bit?

$$\begin{array}{r} \text{5 4 3 2 1 0} \\ | 0 1 1 0 | \\ \times 0 0 0 0 0 1 \\ \hline 0 0 0 0 0 1 \end{array}$$

$0 \& 1 \rightarrow 0$
 $1 \& 1 \rightarrow 1$

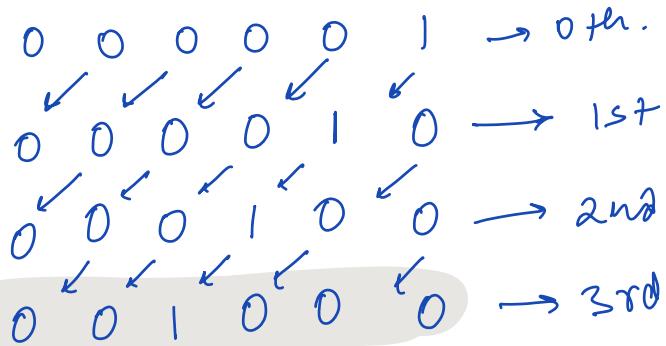
$$\begin{array}{r} \text{5 4 3 2 1 0} \\ | 0 1 1 0 | \\ \times 0 0 1 0 0 0 \\ \hline 0 0 1 0 0 0 \neq 0 \end{array}$$

$$\begin{array}{r} \text{1 0 1 1 0 1} \\ | 0 1) \\ \times 0 0 1 0 0 0 \\ \hline 0 0 0 0 0 0 \Rightarrow 0 \end{array}$$

$$\begin{matrix} 5 & 4 & 3 & 2 & 1 & 0 \\ | & 0 & 1 & 1 & 0 & 1 \end{matrix} \Rightarrow N = 45$$

left shift

$1 \ll i$



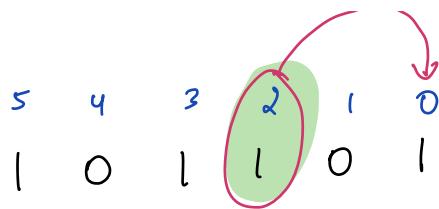
$$N \& (1 \ll i) \rightarrow \begin{cases} 0 & [\text{unset}] \\ !0 & [\text{set}] \end{cases}$$

```
bool checkBit (int N, int i) {
    if (N & (1 << i)) {
        return true
    }
    else {
        return false
    }
}
```

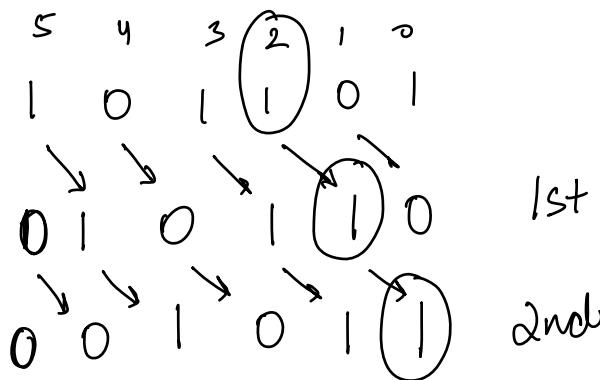
TC $\rightarrow O(1)$
SC $\rightarrow O(1)$

10th

7 5 4 3 2 1 0
 0 1 0 1 1 0 1
i = 10th



0 0 0 0 0 1



$N \gg i$

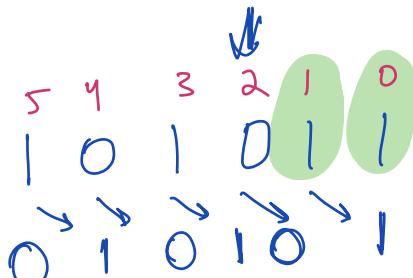
bool checkBit2 (int N, int i) {

if (($N \gg i$) & 1) {
 return true
}

TC \rightarrow O(1)
SC \rightarrow O(1)

else {
 return false
}

N:



cnt =
1
2

Given N, i^o . set i^o bit of N

$$\begin{array}{l} 1 \rightarrow 1 \\ 0 \rightarrow 1 \end{array}$$

$N = 4$ $\begin{array}{ccccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}$

$i = 0$ $\begin{array}{ccccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array} \Rightarrow 5$

$i = 1$ $\begin{array}{ccccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array} \Rightarrow 6$

$i = 2$ $\begin{array}{ccccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \Rightarrow 4$

Return the resulting number.

$$0_1 | 1 \rightarrow 1$$

$$0 | 1 \rightarrow 1$$

$$1 | 1 \rightarrow 1$$

$\begin{array}{ccccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}$

(OR) $\begin{array}{r} 00 | 00000 \\ \hline 00 | 00000 \end{array}$

$$\begin{array}{l} 1 | 0 \rightarrow 1 \\ 0 | 0 \rightarrow 0 \end{array}$$

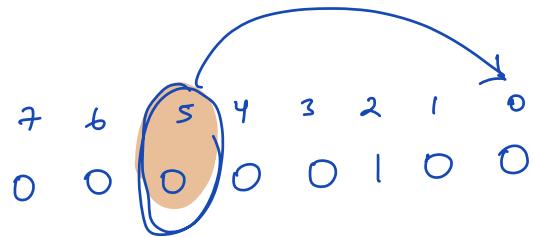
$$\begin{array}{r} 00 | 00100 \\ \hline 00100 \end{array} \Rightarrow 2^5 + 2^2 = 36$$

int setBit(N, i) {

| return $N | (1 \ll i)$

y

$((N \gg i) | 1) \times$



$$\begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & X \end{array}$$

Clearing a Bit

Given N, i ; clear i th bit in N

N : 24

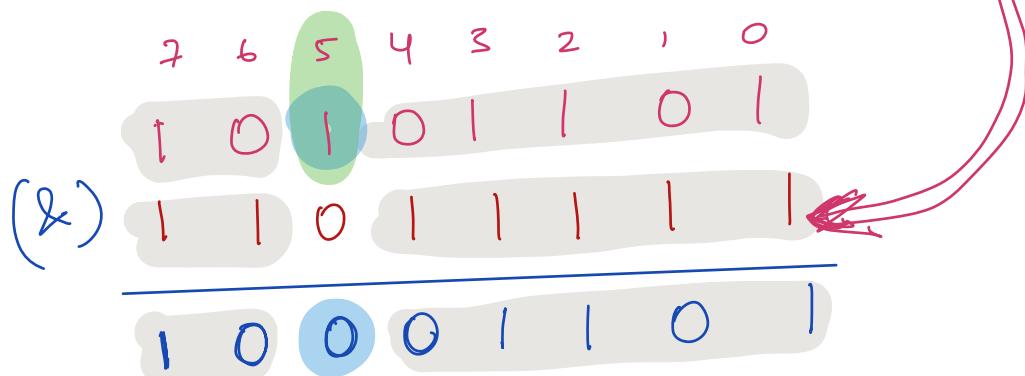
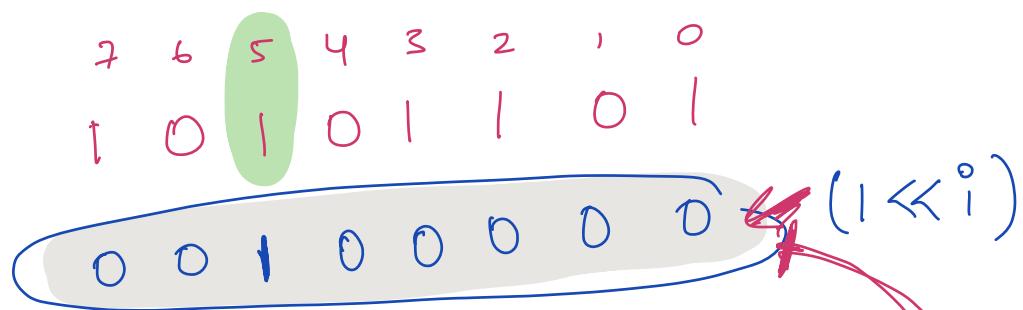
4 3' 2 1 0

1 1 0 0 0 → 24

1 0 0 0 0 → 16.

$1 \& 0 \rightarrow 0$

$0 \& 0 \rightarrow 0$



$$\sim(1 \ll^i) \quad \sim(0 \downarrow 0 \downarrow 1 \downarrow 0 \downarrow 0 \downarrow 0 \downarrow 0 \downarrow)$$

Below the equation, the binary number 11011111 is shown with arrows pointing down to each bit, indicating the result of the NOT operation on the cleared bit position.

```
int clearBit( N, i ) {  
    return N & ~( $1 \ll^i$ )  
}
```

Break → 10:40

$N \wedge (\underline{i \leq i})$

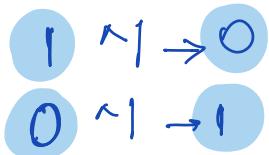
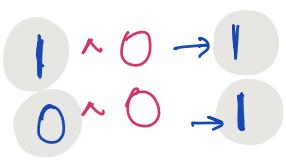
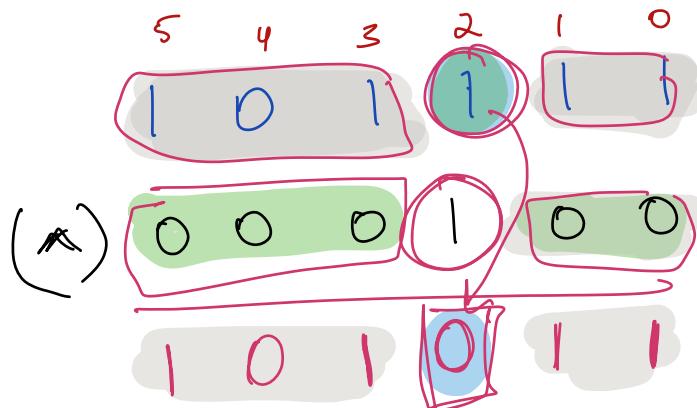
$$\begin{array}{r} 7 \ 6 \ 5 \ 4 \ 3 \ 2 \ 1 \ 0 \\ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \\ \hline 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \\ \hline 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \end{array}$$

$$\begin{array}{r} 7 \ 6 \ 5 \ 4 \ 3 \ 2 \ 1 \ 0 \\ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \\ \hline 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \\ \hline 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \end{array}$$

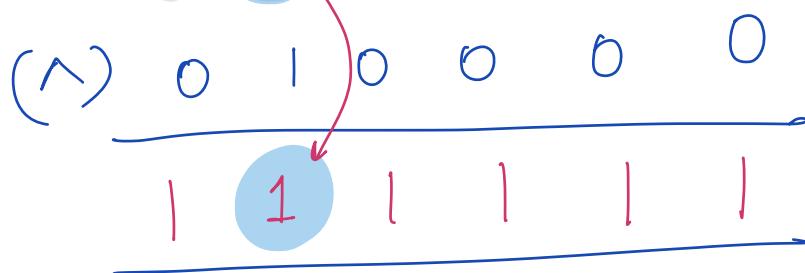
Ques. N, i ; toggle the i^{th} bit

$$N: 47$$

$$i: 2$$



$$N: 47$$



```
int toggleBit (N, i) {
```

```
    return N  $\wedge$  (1  $\ll$  i)
```

```
}
```

Given a ve + no. N , toggle all the bits starting from the rightmost set Bit.

$N = 20$

1 0 1 0 0 \Rightarrow

1 0 0 1 1

$N = \textcircled{24}$

1 1 0 0 0 \downarrow

1 0 1 1 1

$N = 8$

0 1 0 0 0 \downarrow

0 0 1 1 1

$N = 0$

0 0 0 0 0 0 0 0 \downarrow

`if ($N == 0$) return 0 // edge case`

$i = 0$

`while ($(1 \ll i) < N$) {`

`if (checkBit(N, i) == 0) {`

`setBit(N, i)`

`} else {`

`togglebit(N, i)`

`break`

`}
 i++`

`}`

$N = 20$

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

1	0	1	0	0
	0	0	1	1

$\stackrel{TC}{=}$

$O(\log_2 N)$

SC

$O(1)$

Observation

	N	$N-1$	$N \& (N-1)$
3:	0 1 1	0 1 0 $\Rightarrow 2$	0 1 0
5:	1 0 1	1 0 0 $\Rightarrow 4$	1 0 0
15:	1 1 1 1	1 1 1 0 $\Rightarrow 14$	1 1 1 0
20:	1 0 1 0 0	1 0 0 1 1 $\Rightarrow 19$	1 0 0 0 0
24:	1 1 0 0 0	1 0 1 1 1 $\Rightarrow 23$	1 0 0 0 0
44:	1 0 1 1 0 0	1 0 1 0 1 1 $\Rightarrow 43$	1 0 1 0 0 0

```

int toggleByRight {
    return  $N-1$ 
}

```

Google | Amazon

Q: Given an int N , count the no. of set bits present in N .

$N = 44 \quad 1 \ 0 \ 1 \ 1 \ 0 \ 0$

ans: (3) =

```
ans = 0  
while ( $N > 0$ ) {  
    if ( $N \& 1 == 1$ ) ans++  
    N = N  $\gg 1$   
}
```

$N = \frac{N}{2}$

TC
 $O(\log_2 N)$

SC
 $O(1)$

	N	N-1	N & (N-1)
3:	0 1 1	0 1 0 $\Rightarrow 2$	0 1 0
5:	1 0 1	1 0 0 $\Rightarrow 4$	1 0 0
15:	1 1 1	1 1 1 0 $\Rightarrow 14$	1 1 1 0
20:	1 0 1 0 0	1 0 0 1 1 $\Rightarrow 19$	1 0 0 0 0
24:	1 1 0 0 0 0	1 0 1 1 1 $\Rightarrow 23$	1 0 0 0 0
44:	1 0 1 0 0	1 0 1 0 1 $\Rightarrow 43$	1 0 1 0 0 0

$$1 \& 0 \rightarrow 0$$

$$0 \& 1 \rightarrow 0$$

Result of the toggled form is always 0.

$N \& (N-1) \Rightarrow$ rightmost bits are no more.

1 0 1 1 0 1 0

↓ $N \& (N-1)$

1 0 1 1 0 0 0

↓ $N \& (N-1)$

1 0 1 0 0 0 0

↓ $N \& (N-1)$

1 0 0 0 0 0

↓ $N \& (N-1)$

0 0 0 0 0 0

No. of times I'm clearing 1s =
Count of 1s.

$$15 = 1111 \Rightarrow 4$$

$$16 = 10000 \Rightarrow 1$$

$$17 = 10001 \Rightarrow 2$$

$$2^{30} = 1000\ldots0 \Rightarrow 1$$

Previous code

$$\log_2 15$$

$$\log_2 16$$

$$\log_2 17$$

$$\log_2(2^{30}) = 30$$

int countSetBits(N){

```
int ans = 0  
while (N > 0) {  
    N = N & (N-1)  
    ans++
```

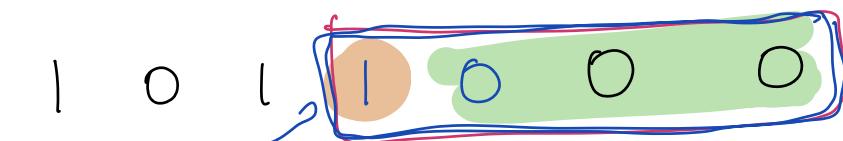
TC = O(log₂N)

SC = O(1)

$$N \rightarrow N-1$$

ans.

6 5 4 3 2 1 0



$$-2^3 + [2^0 + 2^1 + 2^2]$$

$$\cancel{2^5} + \cancel{2^5} = \underline{\underline{-1}}$$

2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	1	0	0	0

\downarrow

0	0	1	0	1	1	1
---	---	---	---	---	---	---

\downarrow

$$-2^3 + [2^0 + 2^1 + 2^2]$$

$$\cancel{-2^5} + \cancel{2^5} = \underline{\underline{-1}}$$

0 0 0 1 0 0

$\downarrow N \times (N-1)$

0 0 0 0 0