→ Left Shift
→ Right Shift
→ Check bit / Count Bit
→ Set bit
→ Unset bit
→ Negative number
→ Significance of MSB
→ Datatypes Ranges.

→ Next Class

Bit manipulation-3

int 32 bites

8 bit number

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
a=10	0	0	0	0	1	0	1	0	$2^3+2^1 = 8+2 = 10$
	7th	6th	5th	4th	3rd	2nd	1st	0th	
a<<1	0	0	0	1	0	1	0	0	$= 2^4+2^2 = 16+4 = 20$
a<<2	0	0	1	0	1	0	0	0	$= 2^5+2^3 = 32+8 = 40$
a<<3	0	1	0	1	0	0	0	0	$= 2^6+2^4 = 64+16 = 80$
a<<4	1	0	1	0	0	0	0	0	$= 2^7+2^5 = 128+32 = 160$
a<<5	0	1	0	0	0	0	0	0	$= 2^6 = 64$

Overflow = Exceeding the size.

General (No overflow)

int x = 10

$a << 1 = a \times 2$

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

$a << 3 = a \times 2^3$

$a << 4 = a \times 2^4$

$a << N = a \times 2^N$.

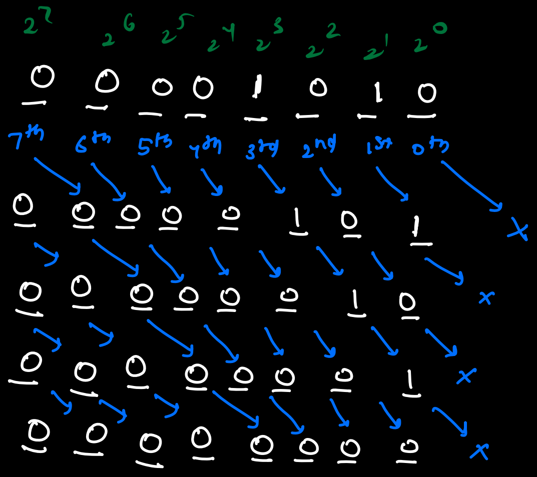
$1 << 1 = 1 \times 2^1 = 2$

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

⋮

$1 << N = 2^N$.

Right shift



a =10 — 0 0 0 0 1 0 1 0

a >> 1 — 0 0 0 0 0 1 0 1

a>>2 — 0 0 0 0 0 0 1 0

a >> 3 — 0 0 0 0 0 0 0 1

a >> 4 — 0 0 0 0 0 0 0 0

$2^3 + 2^1 = 10$

$\downarrow /2$

$2^2 + 2^0 = 5$.

$\downarrow /2$

$2^1 = 2$

$\downarrow /2$

$2^0 = 1$

$\downarrow /2$.

0

Generalise

$$x >> 1 = \frac{x}{2}$$

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

$$x >> 3 = \frac{x}{2^3}$$

$$x >> N = \frac{x}{2^N}$$

int → 4 bytes → 32 bits
[31 ... - 2, 1, 0].

long → 8 bytes → 64 bits
↓
[63 ... 0]

$$\begin{array}{cccccccccccccc} \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ \\ 31 & . & . & . & . & . & . & . & . & . & 3 & 2 & 1 & 0 \end{array}$$

Q. Given N, i. Check if i^{th} bit pos in N is set or not

N = 21
i = 2

$$\begin{array}{ccccc} 4 & 3 & \boxed{2} & 1 & 0^{th} \\ 1 & 0 & \boxed{1} & 0 & 1 \end{array}$$

Bit is set
ans = True

N = 34
i = 3

$$\begin{array}{cccccc} 5 & 4 & \boxed{3} & 2 & 1 & 0 \\ 1 & 0 & \boxed{0} & 0 & 1 & 0 \end{array}$$

Bit is unset
ans = False.

Given N, 0^{th} bit set or not

```
If ( N & 1 == 1 ) return True
else return False.
```

```
boolean checkbit ( int N, int i )
{
        if ( (N>>i) & 1 == 1 )  return True
        else  return False
}
```

```
boolean checkbit ( int N, int i )
{
     return (N>>i) & 1 == 1
}
```

N = 34
i = 3

```
5 4 3 2 1 0          5 4 3 2 1 0
1 0 0 0 1 0          1 0 1 0 1 0
0 0 1 0 0 0          0 0 1 0 0 0
-----------          -----------
0 0 0 0 0 0          0 0 1 0 0 0
```

```
bool checkbit ( int N, int i )
{
      if ( (N & (1<<i)) == 0 )  return False
      else  return True.
}
```

```
bool checkbit( int N, int i)
{
                                 != 0
    if( (N & (1<<i)) == (1<<i) )        return True
    else   return False
}
```

```
bool checkbit( int N, int i)
{                                                          O(1)
    return !( (N & (1<<i)) == 0)
    return (N & (1<<i)) == (1<<i)
    return (N & (1<<i)) != 0.
}
```

Q. Given a number N. (integer.) Cal how many bits are set.

N = 10	1010	ans = 2.
N = 27	11011	ans = 4

Idea 1

32 iterations

```
cnt = 0.
for( i=0 ; i ≤ 31 ; i++ )
{
    if ( checkbit( N, i) ) cnt++
}
return cnt
```

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

idea 2.

```
N=10      1 0 1 0      N&1 ==0
           ↘ ↘ ↘
N>>1      0 1 0 1      N&1== 1      cnt=1

N>>2      0 0 1 0      N&1==0

N>>3      0 0 0 1      N&1 ==1      cnt=2

N>>4      0 0 0 0
```

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

TC: O(logN)
SC: O(1).

Break

10:24 10:32.

N is capped to 2^{32}

↑

$O(\log N)$ approach better than

$O(1)$ approach.

↓

32 iterations.

Q. Given N, set the i^{th} bit.

$$N = 10 \quad \begin{array}{cccc} 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{array} \longrightarrow 1110 \qquad \text{ans} = 14.$$

$i = 2$

$N = 10$ $\quad 1010 \longrightarrow 1011 \longrightarrow$ ans = 11.

$i = 0$

```
int setbit ( int N, int i)
{
    If ( check bit (N, i)) return N.
    else return N + 2^i
}
```

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

$$\begin{array}{lcccccccccc}
 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
 & 1 & 0 & 1 & \times & 1 & 1 & 1 & 0 & 0 & 1 \\
\cdot\ \text{OR} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1
\end{array} \longrightarrow 2^i \quad (1 << i)$$

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

Q. Given N, i. Unset ith bit.

```
          4 3  2 1  0
N =       1 0  1 1  0
i=2            ↕
       ____________________
          1 0  0 1  0  → ans.
```

```
boolean unset(N, i)                              2^i = 1<<i
{
        If ( check(N, i)) return N - 2^i
        else return N
}
```

```
     1 0 1 0/1  0 1 0          & 0
&    1 1 1 0    1 1 1          ~(0001000)
    ___________________
     1 0 1 0    0 1 0
```

```
boolean unset(N, i)
{
        return N & (~(1<<i))
}
```