Lecture :- Bit Manipulation - 1

Agenda

- Basics
- Making sure you are comfortable with bits.

Class start at 8:35 PM

Decimal no. system [10] [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Examples: 582, 4089, 700, 1 etc

Binary no. system [2] [0, 1]

Examples: 00, 100, 1010110, 89 ✗

1010110 — Binary / Decimal

Conversion

Q. **convert decimal to binary.**

1.) $(20)_{10} = (x)_2$

2	20	
2	10	0
2	5	0
2	2	1
	1	0

$(10100)_2$.

2.) 45.

2	45	
2	22	1
2	11	0
2	5	1
2	2	1
	1	0

$(101101)_2$.

Q. Convert binary to decimals.

$(101101)_2 = (45)_{10}$

$\Rightarrow (\overset{5\ 4\ 3\ 2\ 1\ 0}{1\ 0\ 1\ 1\ 0\ 1})_2$

$\Rightarrow 1*2^5 + 0*2^4 + 1*2^3 +$

$1*2^2 + 0*2^1 + 1*2^0$

$\Rightarrow 32 + 0 + 8 + 4 + 0 + 1$

$\Rightarrow 45.$

$\overset{3\ 2\ 1\ 0}{1\ 2\ 3\ 4}$

th. hun ten unit

$\Rightarrow 1*10^3 + 2*10^2 +$

$3*10^1 + 4*10^0.$

Addition

Decimals:

$$\begin{array}{r} {\scriptstyle 1\ \ 1\ \ } \\ 238 \\ 469 \\ \hline 707 \end{array}$$

Binary

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 2\ [10]$

$1 + 1 + 1 = 3\ [11]$

$$\begin{array}{r} {\scriptstyle 1\ \ 1\ \ 1\ \ } \\ 0101 \\ 0011 \\ \hline 1000 \end{array}$$

$$\begin{array}{r} {\scriptstyle \ \ 1\ \ 1\ \ } \\ 10110 \\ 00111 \\ \hline 11101 \end{array}$$

Bitwise operators

And	or	xor	not	left shift	right shift
&	|	^	~ !	<<	>>

Same same

A	B	A & B (0 dominates)	A | B (1 dominates)	A ^ B (pupp shame)
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Qu 5 & 6. ⇒ 4 ⇒ 5 ⇒ 1 0 1

6 ⇒ & 1 1 0

2 1 0

1 0 0 ⇒ $(4)_{10}$

$2^2 + 0 + 0$

Qu 6 & 11 = 2

6 = 0 1 1 0

11 = & 1 0 1 1

3 2 1 0

$(0\ 0\ 1\ 0)_{(2)} = (2)_{10}$

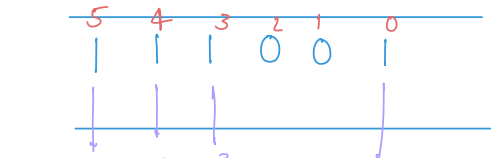
$0*2^3 + 0*2^2 + 1*2^1 + 0*2^0$

$0 + 0 + 2 + 0 = 2$

Qu $20 \wedge 45 = (57)_{10}$

20 = 0 1 0 1 0 0

45 = ^ 1 0 1 1 0 1



$1*2^5 + 2^4 + 2^3 \qquad 2^0 = 32 + 16 + 8 + 1 = 57$

Qu Given a no. check whether it is even or odd?

```
String isEvenOrOdd(int n) {
    if (n % 2 == 0) {
        return "Even";
    }
    return "odd";
}
```

Constraint: Do not use % operator

Observation 1

16 — 10000
14 — 1110
26 — 11010

} Even no :- Rightmost bit = 0

15 — 1111
17 — 10001
9 — 1001
7 — 111

} Rightmost bit = 1.

observation 2

n = 126

& 1.

```
  0 1 1 1 1 1 1 0
& 0 0 0 0 0 0 0 1
-----------------
  0 0 0 0 0 0 0 0   = (0)10
```

n = 31

& 1

```
  1 1 1 1 1
& 0 0 0 0 1
-----------
  0 0 0 0 1   = (1)10
```

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

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

Conclusion [property]

n & 1

- Even → 0 (if n rightmost bit is 0)
- odd → 1 (if n rightmost bit is 1)

Algorithm

```
string evenOrOdd(int n) {
    if(n & 1 == 0) {
        return "Even";
    } else {
        return "odd";
    }
}
```

internally

Approach3 (Kaviarasu).

$n = \frac{26}{2} = 13 * 2 = 26$

$26 == 26$ [Even]

$n = \frac{27}{2} = 13 * 2 = 26$

$26 != 27$ [odd]

Break :- 9:54 PM

Commutative property

$a \& b = b \& a$

$a | b = b | a$

$a \wedge b = b \wedge a.$

Associative property

$a \& b \& c = (a \& b) \& c = a \& (b \& c)$

$a | b | c = (a | b) | c = a | (b | c)$

$a \wedge b \wedge c = (a \wedge b) \wedge c = a \wedge (b \wedge c).$

Q 1) $n \& 0 = 0$ [$1234 \& 0$]

Ex: $28 \& 0$

$28 = 11100$

$\& \ 00000$

$(00000)_2 = (0)_{10}$

2) $n \& n = n.$

$n = 124.$

$124 \& 124$

$124 = 1111100$

$124 = \& \ 1111100$

1111100

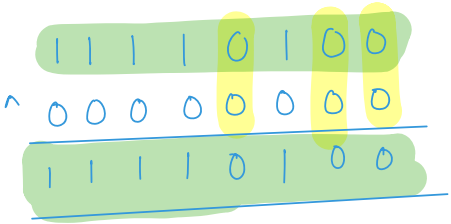
3.) $n \mid 0 = n$. [1 is dominating]

no significance.

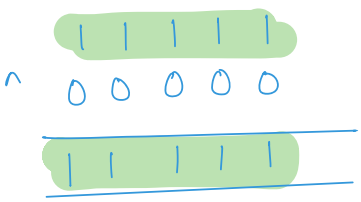
4.) $n \mid n = n$.

5.) $n \wedge 0 = n$.

$n = 124$



$n = 31$



6.) $n \wedge n = 0$

Qu Amazon, Adobe, Microsoft

Given arr[n], every element appears twice except for one element which appear once. find that unique element?

arr[5] = [6, 9, 9, 10, 6] = 10

arr[7] = [6, 3, 6, 5, 3, 4, 4] = 5 Ans

Brute | obvious approach

```
int uniqueElement(int[] arr) {
    for(int i=0; i<arr.length; i++) {
        int currEl = arr[i];
        int cnt = 0;
        // find its freq in arr
        for(j=0; j<arr.length; j++) {
            if(arr[j] == currEl) {
                cnt++;
            }
        }
        if(cnt == 1) {
            return currEl;
        }
    }
    return -1; // No el in arr with freq 1.
}
```

TC: $O(n^2)$

SC: $O(1)$

Approach2: Sorting

arr[7] = [6 3 6 5 3 4 4]

↓ sort (nlogn)

arr[7] = [3 3 4 4 5 6 6] → O(n)

0 2 4

arr[0] == arr[1] → 2

arr[2] == arr[3] → 4

arr[4] != arr[5] → ans = arr[4]

TC := O(n logn) + O(n) ~ O(nlogn)

SC: O(1).

Approach3 Hashmap. → TC: O(n)

SC: O(n).

Constraint: TC = O(n)

SC ≃ O(1).

Approach4: $n \wedge n = 0$

arr[7] = [6 3 6 5 3 4 4]

$\Rightarrow 6 \wedge 3 \wedge 6 \wedge 5 \wedge 3 \wedge 4 \wedge 4$

$\Rightarrow (6 \wedge 6) \wedge 3 \wedge 3 \wedge 4 \wedge 4 \wedge 5$

$\Rightarrow (0 \wedge 3) \wedge 3 \wedge 4 \wedge 4 \wedge 5$

$\Rightarrow (3 \wedge 3) \wedge (4 \wedge 4) \wedge 5$

$\Rightarrow 0 \wedge 0 \wedge 5 = 5$ Any

Observation

$1 \wedge 2 \wedge 1 \wedge 2 \wedge 3 \Rightarrow 3$ Any

$1 \wedge 2 = 0\,1 \wedge 1\,0 = 1\,1 = 3$

$3 \wedge 1 = 1\,1 \wedge 0\,1 = 1\,0 = 2$

$2 \wedge 2 = 0$

$0 \wedge 3 = 3$

Property: $a \wedge b \wedge c \wedge d \wedge e$

(or)

$a \wedge d \wedge e \wedge b \wedge c$

(or)

$d \wedge a \wedge c \wedge b \wedge e$

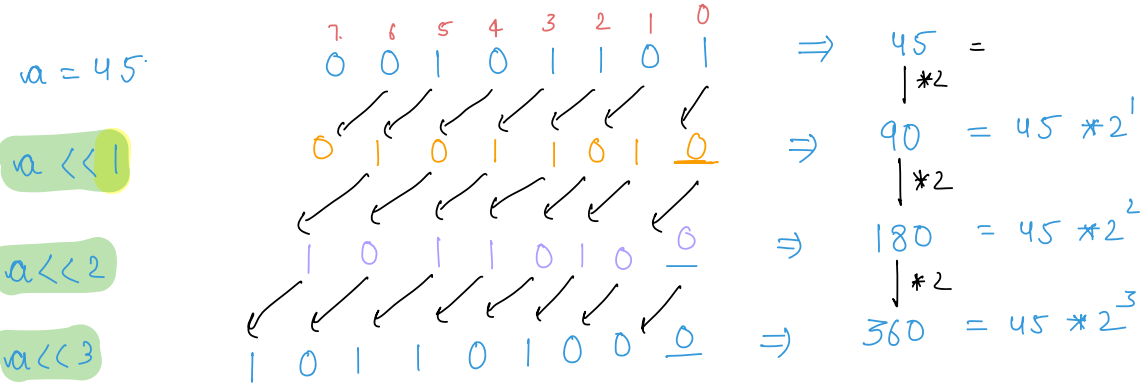
same { order does not matter for xor }

```
int uniqueElement(int[] arr) {
    int ans = 0;
    for(i=0; i<arr.length; i++){
        ans = ans ^ arr[i];
    }
    return ans;
}
```

TC: O(n)

SC: O(1)

* **Left shift (<<) operator**



Property

a

$a << 1 = a * 2^1$

$a << 2 = a * 2^2$

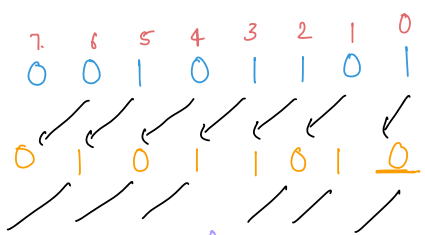
$a << 3 = a * 2^3$

$a << n = a * 2^n$

proof:

$a = 45$

$a << 1$



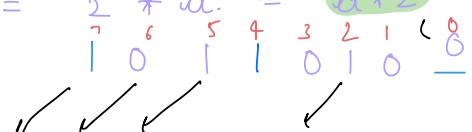
$a \rightarrow 2^5 + 2^3 + 2^2 + 2^0$

$a<<1 \rightarrow 2^6 + 2^4 + 2^3 + 2^1$

$= 2(2^5 + 2^3 + 2^2 + 2^0)$

$= 2 * a = a * 2^1$

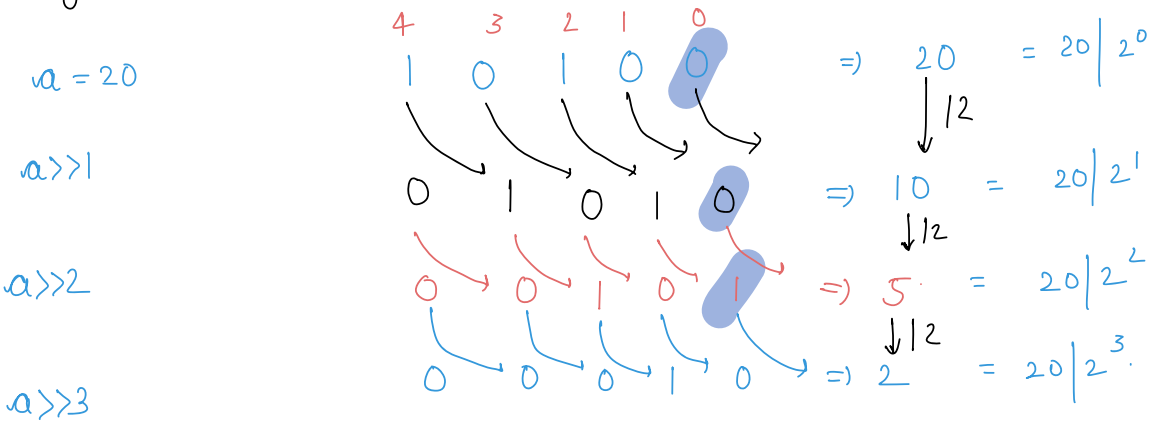
$a<<2 =$



$2^7 + 2^5 + 2^4 + 2^2$

$2^2(2^5 + 2^3 + 2^2 + 2^0) = 2^2 * a$

Right shift (>>)



Property:

a

$a >> 1 \rightarrow \frac{a}{2^1}$

$a >> 2 \rightarrow \frac{a}{2^2}$

$\vdots$

$a >> n \rightarrow \frac{a}{2^n}$

Thank you

Doubts

int a = Integer.MAX_VALUE

int c = a<<1. (wrong ans).

long c = a<<1. (Right ans).

int c = Integer.MAX * Integer.MAX

int c = $10^5 * 10^5 \neq 10^{10}$