Day 10
20/07/25

Binary Number System

Topics : Binary Numbers System, Converssion code of Binary to Decimal, Decimal to Binary 1's Complement 2's Complement

---

Math : Decimal Number System Base 10

Dec
0 to 9 (10 digits)

The Word "Dec" means 10 "The name decimal Numbers System is given Bcz it have 10 digits"

hexadecimal

hexa = 6 dec = 10
10 + 6 = 16 Base 16

Binary Numbers System

The Number System which have Base 2 and It's also computer understandable Numbers System.

The computer understands only two states 0, 1 which have Base 2

octal

8 digits

Numbers System	Base
Decimal	10
Hexa decimal	16
octal	8
Binary	2

Binary Numbers System and Converstions

42 to convert to Binary we have to perform repeated division with 2

divisor	divident	remainder
2	42	0
2	21	1
2	10	0
2	5	1
2	2	0
2	1	1
	0	

1 0 1 0 1 0

Binary form of 42

$$(42)_{10} = \underbrace{(101010)_2}_{\text{Base}}$$

2	50	0
2	25	1
2	12	0
2	6	0
2	3	1
2	1	1
	0	

$$(110010)_2 = (50)_{10}$$

Decimal	Binary
1	1
2	010
3	011
4	100
5	101
6	110
7	111
8	1000
9	1001
0	0000
10	1010

2	8	0
2	4	0
2	2	0
2	1	1
	0	

2	9	1
2	4	0
2	2	0
2	1	1
	0	

Code for Converting Decimal Number to Binary

```
ans = 0   // Binary Number Variable
pow = 1
while (decNumber > 0) {

    remainder = decNumber % 2 ;
    decNumber = decNumber / 2 ;
    ans = ans + (remainder * power) ;
    pow = pow * 10 ;

}
Print (ans)
```

Suppose for Example dec = 5

2	5	1
2	2	0
2	1	1
	0	

---

5 can Be represented as

$$1 \times 10^2 + 0 \times 10^1 + 1 \times 10^0$$
$$100 + 0 \times 10 \quad 1 \times 1$$
$$100 + 0 + 1$$
$$\underline{101}$$

Initially power is set to 1 and after Each iteration the power is multiplied with 10 and multiplied with remainder and added to ans

---

Binary To Decimal

10 1 0 1 0

2^{pow}

$$2^5 \times 1 + 2^4 \times 0 + 2^3 \times 1 + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 0$$
$$32 + 0 + 8 + 0 + 2 + 0$$
$$Dec = 42$$

1 1 0 0 1 0

$0 \times 2^0 = 0 \quad +$
$1 \times 2^1 = 2 \quad +$
$0 \times 2^2 = 0 \quad +$
$0 \times 2^3 = 0 \quad +$
$1 \times 2^4 = 16 \quad +$
$1 \times 2^5 = 32 \quad +$

Dec 50

1100101

$64 + 32 + 2^2 + 1$

$64 + 32 + 4 + 1$

dec = 101 (one hundred one)

Binary To Decimal

```
binNum ;   ans = 0 ;   power = 1   // 2^power

while ( binNum > 0) {

    remainder = binNum % 10 ;
    ans = ans + remainder * power ;

    binNum = binNum / 10 ;

    power = pow * 2 ;
}

print (ans) ;
```

2^{power}	2^5	2^4	2^3	2^2	2^1	2^0
	32	16	8	4	2	1
25 =	0	1	1	0	0	1

25 = 11001

Position of 2^{power} matters

NOTE / TIP

When we convert any odd Number to Binary last digit is 1

3 011
5 101
7 111

Binary Number addition

Addition In Binary

0 + 0 = 0
1 + 0 = 1
0 + 1 = 1
1 + 1 = 1 carry + 0

1+1=2

Binary of 2
10 = dec(2)

0	0	0
0	1	1
1	0	1
1	1	10

Two's Complement (-ve number Binary form)

int n = -10 (-ve Ten)

Binary 10 = 01010

Least Significant Bit

Most Significant Bit

int 4 Byte = 32 Bit

32 Bit

most Significant Bit
MSB

Least Significant Bit
LSB

most important Bit is MSB
which tells the sign of a Number
-ve or +ve

Method to take 2's complement

1. Convert the Number to Binary
2. Prefix with 0 (MSB is 0) if MSB == 0 Then Number is +ve
3. 1's complement

   0 → 1
   1 → 0

4. add 1

Two's Complement of n = 10

Step 1   1010   Binary Value of 10

Step 2   01010   MSB = 0

Step 3   1's Complement

10101

Step 4   add 1 to 10101

$$\begin{array}{r} 10101 \\ +1 \\ \hline 10110 \\ \hline \end{array}$$

Decimal value of $(-10) = (10110)_2$

$\boxed{10110}$ MSB = 1   -Ve Number

Reverse of Two's Complement of -Ve Number

MSB is 1

Bin → Dec

2's complement

Step 1   1's complement

Step 2   +1

Binary - 10110   int n = -10

Binary = 10110

step 1 1's Complement

01001

step 2 add + 1

01010

1010 = 10

we know that MSB of −10 is 1

(1010) = $(10)_{dec}$

= −10 //

Convert −8 to Binary

1000 = $(8)_{10}$

−8

step 1 Binary of 8 with MSB 0

01000

1's Complement

10111

step 2

add 1 + 1

11000

11000

MSB −ve 8

$\boxed{-8}$

11000 to Dec

MSB

11000

1's complement

00111

+ 1

01000

dec = 8

MSB is 1 so

dec = −8

int −12

MSB 01100

1's Complement

10011

+1

10100

10100

1's

01011

+1

01100

−12

Bit Manipulation

Topic :- Bitwise Operators AND OR NOT Right & left Shift

Bit wise operators

among two Bits

1. AND — if Both Bits are 1 then output is 1 Else 0
2. OR — if any Bit is 1 then output is 1 Else 0
3. XOR — if Both Bits are not same output 1 Else 0
4. Right Shift — shifting Bits towards Right
5. left Shift — shifting Bits towards left
6. NOT — Compliment of 0, 1

AND

x	y	x & y
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	x | y
0	0	0
0	1	1
1	0	1
1	1	1

XOR

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

NOT

0	1
1	0

Right Shift >>

Basically Right Shift is Equal to division By 2

Example take Decimal Value 13

13 = 1 1 0 1

0 0 | 1 | 1 | 0 | 1 |

Right shift 13 >> 1 13 right shift By 1 Bit

0 0 | 1 | 1 | 0 | 1 |

| 0 | 1 | 1 | 0 | 1

13 >> 1 Integer division of $\frac{13}{2} = 6$

13 >> 1 = 0 1 1 0

= 6

---

13 >> 2 13 right shift By 2

0 0 0 | 1 | 1 | 0 | 1 |

| 0 | 0 | 1 | 1 | 0 1

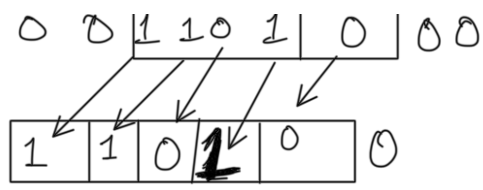
$13 >> 2 = \frac{13}{2^2} = \frac{13}{4} = 3$

$\boxed{13 >> 2 = 3}$

$\boxed{n >> k = n/2^k}$

Left shift <<

13 << 1



$13 * 2^1$

$26 \Rightarrow$ | 1 | 1 | 0 | 1 | 1 | 0 |

Basically In simple words left shifting means multiply Numbers in 2Powers

$$n << k = n * 2^k$$

NOT

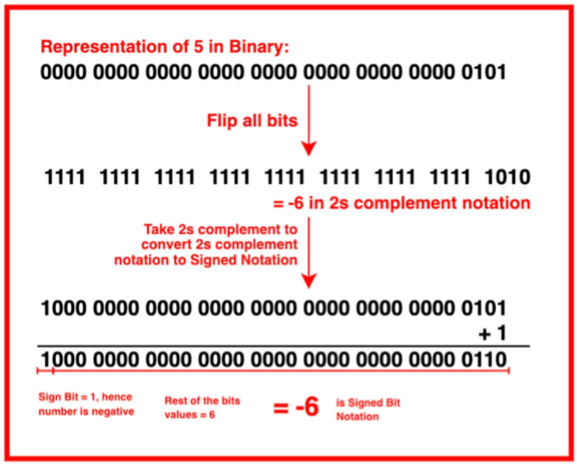
Eg: $0101 = 5$

$\sim 5 = \sim 0101$

```
  0101
    +1
  ----
  0110
```

MSB=1

$$\boxed{\sim 5 = -6}$$



BIT MANIPULATION Questions

Set Bit = 1   reset Bit = 0

toggle $0 \to 1$, $1 \to 0$ clear $1 \to 0$, $0 \to 0$

Next session diving deeper into Bit Manipulation

Questions

1. Remove the Rightmost set Bit
2. Question on power set / Bit manipulation
3. Swap two Numbers without using temp
4. Count total set Bit
5. Toggle the 'i'th Bit
6. 1's and 2's Compliment

(7) Divide Two integers without using multiplication and division operators

(8) XOR of Numbers In a Given Range

(9) In an array Exactly two elements appear only once and all other elements appear Exactly twice return the two Elements that appear only once.

(10) In an array Every element appears thrice Except for one Element Return the Element that only appears once

(11) Given an array of numbers, where Every number appears twice Except for one element Return the element that only appears once.

(12) Minimum Bit flips to convert a Number

(13) Clear the 'i'th Bit

(14) Set the 'i'th Bit

* learn How Int Datatypes Stored In Memory

(15) Counting Bits (i) count set Bits (ii) Count unset Bits

(16) Reverse Bits

(17) Check if k^{th} Bit is Set or Not

(18) Check weather the given Number is Even or odd

(19) Count total set Bits in numbers from 0 to N