**Day – 30 of the 101 days of coding challenge**

**Problem:**

The **complement** of an integer is the integer you get when you flip all the 0's to 1's and all the 1's to 0's in its binary representation.

* For example, The integer 5 is "101" in binary and its **complement** is "010" which is the integer 2.

Given an integer n, return *its complement*.

**Example 1:**

**Input:** n = 5

**Output:** 2

**Explanation:** 5 is "101" in binary, with complement "010" in binary, which is 2 in base-10.

**Example 2:**

**Input:** n = 7

**Output:** 0

**Explanation:** 7 is "111" in binary, with complement "000" in binary, which is 0 in base-10.

**Example 3:**

**Input:** n = 10

**Output:** 5

**Explanation:** 10 is "1010" in binary, with complement "0101" in binary, which is 5 in base-10.

**Constraints:**

* 0 <= n < 109

**Code:**

int bitwiseComplement(int n) {

       int num = n;

       int mask = 0;

       if(n == 0) // if n = 0

           return 1;

        while(num!=0){

            mask = (mask<<1) | 1;

            num = num>>1;

        }

        int ans = (~n) & mask;

        return ans;

        }

**Working Procedure-**

Suppose value n = 5 -> 0000000101, last 3 value rev-> 010 (2)

If ~n = ~5 => 111111111—1010

Created mask => 00000000000111 after applying & ->000000010 (got the answer)

Now creating the mask—

Mask = 00000000000 left shif—

1st -> 0000000000(as many time will shift will give only 0)

So with or 1 operator applying(0|1) => 1

Accordingly----

1st -> (mask<<1) | 1 => 0000000010

2nd -> 0000000110

3rd-> 0000000111 (got the desired value)