**LC#191:NO OF 1 BITS**

**Given a positive integer n, write a function that returns the number of set bits in its binary representation (also known as the**[**Hamming weight**](http://en.wikipedia.org/wiki/Hamming_weight)**).**

**Example 1:**

**Input: n = 11 -> 1011**

**Output: 3**

public class Solution {**//ONLY FOR POSITIVE TC:O(32) SC:O(1)**

    public int hammingWeight(int n) {

        int count = 0;

        for(int i = 0 ; i<32;i++)

        {

            if((n&1)==1) // checks if the current bit ==1

                count++;

            n=n>>1; //shifts rightly

        }

        return count;

    }

}

**APPROACH 2:**

class Solution {

    public int hammingWeight(int n) {

        int count = 0 ;

        while(n!=0)

        {**//WORKS FOR ALL**

            count = (n&1)+count;

            n=n>>>1;(**HANDLES NEGATIVES)->UNSIGNED RIGHT SHIFT**

        }

        return count;

    }

}

**APPROACH 3:**

class Solution {**//WORKS EVERYWHERE**

public int hammingWeight(int n) {

int count = 0 ;

while(n!=0)

{

**n = n&n-1;**

count++;

}

return count;

}

}