**Bit Manipulation**

Links: https://codesam.blogspot.com/2011/03/

It refers to the process of manipulating individual bits in a binary representation of data.

AND (&) - It performs the AND operation between the corresponding bits of two integers.

if both bits are 1 result is 1 otherwise 0.

0 1 = 0

1 0 = 0

1 1 = 1

0 0 = 0

OR (|) - It performs the OR operation between the corresponding bits of two integers.

if both bits are 0 result is 0 otherwise 1.

0 1 = 1

1 0 =1

1 1 =1

0 0 = 0

XOR (^) - It performs the XOR operation between the corresponding bits of two integers.

if both bits are different result is 1 otherwise it is 0.

1 0 = 1

0 1 = 1

1 1 = 0

1. 0 = 0

NOT(~) - ( Inverts) 1 bit becomes 0 and 0 bit becomes 1

1 -->0

0 -->1

Left Shift (<<) - Moves the bits of a number to left by a specified number of positions.( multiply by 2 raised to that power)

Right Shift (>>) - - Moves the bits of a number to right by a specified number of positions (division by 2 raised to that power)

Count set bits (Strategy: Bit Manipulation)

Count Set - Counting the set bits (1 bit ) in an integer.

1 - set-bit

0 - unset-bit.

package BitwiseAlgorithms;  
  
import java.util.Scanner;  
  
public class CountSetBit1  
{  
 public static int brian(int x)  
 {  
 int r =0;  
 while (x > 0)  
 {  
 x = x & (x-1);  
  
 r++;  
  
 }  
 return r;  
 }  
 public static int countsetbit(int x)  
 {  
 int r =0;  
 while (x > 0) {  
 if ((x & 1)== 1) {  
 r++;  
 }  
 x>>=1;  
 }  
 return r;  
  
 }  
 public static void main(String args[])  
 {  
 int n;  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter an integer");  
 n = sc.nextInt();  
 System.*out*.println("Binary representation of n is : "+ Integer.*toBinaryString*(n));  
 System.*out*.println("Count Set Bit with simple bit manipulation (right shift) is : "+ *countsetbit*(n));  
  
 System.*out*.println("Count Set Bit with Briain bit manipulation is : "+ *brian*(n));  
 }  
}