BITWISE AND LOGICAL COMPLEMENT

1. BITWISE COMPLEMENT (~):

The bitwise complement operator is a unary operator (works on only one operand). It takes one number and inverts all bits of it. When bitwise operator is applied on bits then, all the 1's become 0's and vice versa. The operator for the bitwise complement is ~ (Tilde).

Table 1: Bitwise Not

x	~ x
0	1
1	0

For example,

The bitwise complement operator should be used carefully. The result of ~ operator on a small number can be a big number if the result is stored in an unsigned variable. And the result may be a negative number if the result is stored in a signed variable (assuming that the negative numbers are stored in 2's complement form where the leftmost bit is the sign bit).

Input:
n = 2
Binary form of 2 = 0010
Bitwise complement operation on 2 = ~ 0010
= 1101

1101 is equivalent to decimal value 13.

Expected output: 13
Correct Output : -3

The compiler returns the 2's complement of the input value.

Below shows the steps to find the bitwise complement of binary 2:

Step 1: 2 in binary is **0010**.

Step 2: Apply the bitwise complement operation resulting in **1101**.

Step 3: Interpret the result in two's complement representation: In a 4-bit representation, the leftmost bit (most significant bit) represents the sign. If the sign bit is 0, the number is positive; if the sign bit is 1, the number is negative.

In this case, the leftmost bit is 1, indicating that the number is negative. To determine its decimal value, we calculate the two's complement:

Step 4: Convert the binary representation to decimal: To obtain the decimal value of the two's complement representation, we first invert all the bits (bitwise complement), resulting in **0010**. Then, we add 1 to the resulting value:

0010 + 1 = 0011.

The decimal value of **0011** is -3.

Therefore, using a 4-bit representation, the bitwise complement of 2 is indeed -3.

2. LOGICAL COMPLEMENT (!):

In Java, the logical complement operation is performed using the logical NOT operator (!). The logical NOT operator negates a Boolean value, resulting in the opposite Boolean value. This is a unary operator and returns true when the condition under consideration is not satisfied or is a false condition. Basically, if the condition is false, the operation returns true and when the condition is true, the operation returns false.

Table 2: Logical Not

x	! x
true	false
false	true

For example,

a = 10, b = 20

!(a<b) // returns false

!(a>b) // returns true