**Bitwise & shift operators**

Bitwise Operators are used to perform operations on **bits (0 & 1).**

**Note:** These operators are rarely used in web development, but it’s useful in **cryptography**

**How bitwise do operations?**

1. Bitwise operators **convert** the given decimal number(s) to their binary equivalent number.
2. Then they **perform** the operations on bits of those binary equivalent number(s).

**Bitwise are:**

|  |  |  |
| --- | --- | --- |
| **Bitwise operators** | **Sign** | **Description** |
| And | & | If both Operand 1 and Operand 2 are 1 then the result will be 1, in all other cases result will be 0 |
| OR | | | If both Operand 1 and Operand 2 are 0 then the result will be 0, in all other cases result will be 1 |
| EXCLUSIVE (XOR) | ^ | If both Operand 1 and Operand 2 are same then the result will be 0, otherwise the result will be 1 |
| COMPLEMENT (NOT) | ~ | To the given operand add 1 and change the sign |
| LEFT SHIFT | << | Shifts the bits of first binary number to the left by number of positions indicated by second number |
| RIGHT SHIFT | >> | Shifts the bits of first binary number to the right by number of positions indicates by second number |
| ZERO-FILL RIGHT SHIFT | >>> |  |
|  | | bit 1 means **true** while bit 0 means **false** |

**Important tables:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **&** | | | **|** | | | **^** | | |
| **Operand 1** | **Operand 2** | **result** | **Operand 1** | **Operand 2** | **result** | **Operand 1** | **Operand 2** | **result** |
| **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** |
| **1** | **0** | **0** | **1** | **0** | **1** | **1** | **0** | **1** |
| **0** | **1** | **0** | **0** | **1** | **1** | **0** | **1** | **1** |
| **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

**Example 1: on &, |, ^**

****

**Example 2:**

****

**Illustration: in case of numbers > 1**

**To solve operation we follow these steps:**

1. Convert Operand 1 from decimal to binary system. Use this site to convert numbers <https://decimaltobinary.pro/6>
2. Convert Operand 1 from decimal to binary system.
3. Calculate the result with the help of above table.
4. Convert the result from binary to decimal.

**For example: 6 & 12**

1. **Convert Operator 6 🡺 0110**
2. **Convert Operator 12🡺 1100**
3. **Result 🡺 0100**
4. **100 in binary = 4 in decimal.**

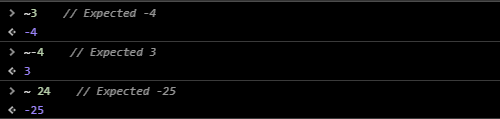
**For example: 6 | 3**

1. **Convert Operator 6 🡺 0110**
2. **Convert Operator 3 🡺 0011**
3. **Result 🡺 0111**
4. **111 in binary = 7 in decimal.**

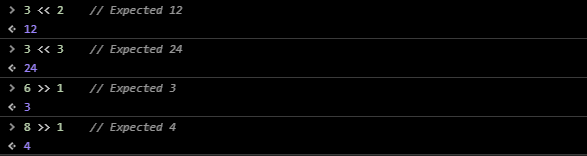
**For example: 6 ^ 2**

1. **Convert Operator 6 🡺 0110**
2. **Convert Operator 2 🡺 0010**
3. **Result 🡺 0100**
4. **11 in binary = 4 in decimal.**

**Example 3:** on Bitwise complement operator

****

**Example 4: on bitwise Left & left Shift operator**

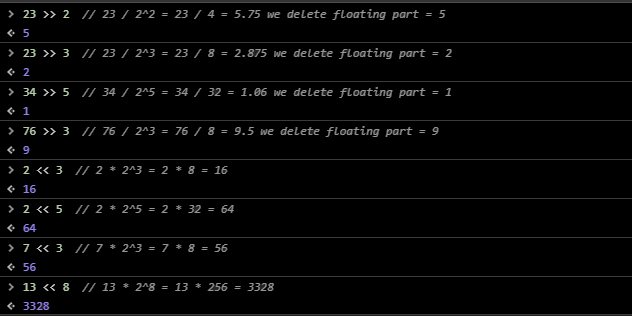
****

**There are equations to solve the last problems:**

1. **In case of left shift operator << : first number \* 2 second number**
2. **In case of right shift operator >> : first number / 2 second number**

**Note: if the result has a floating part we delete it.**

**Examples:**

****

**Note:** operand 1 and operand 2 **always** be **integer** numbers as these integer numbers converted to binary numbers during executing the math operation. And we **can’t ever** convert **fraction decimal** number to binary number. As shown In this picture:

