

## Bitwise Operators

- C Offers bitwise logical operators and shift operators.
  - Look something like the logical operators you saw earlier but are quite different.
  - Operate on the bits in integer values
- Not used in the common program
- One major use of the bitwise AND , &, and the bitwise OR, |, is in operations to test and set individual to test and set individual bits in an integer variable.
  - Can use individual bits to store data that involve one of two choices .
- You could use a single integer variable to store several characteristics of a person.
  - Store whether the person is male or female with one bit.
  - Use three other bits to specify whether the person can speak French , German , or Italian.
  - Another bit to record whether the person's salary is \$50 000 or more.
  - In just four bits you have a substantial set of data recorded.

## Binary Number.

- A binary number is a number that includes only ones and zeroes.
- The number could be of any length/
- The following are all examples of binary numbers:
  - 0                    10101
  - 1                    0101010
  - 10                   1011110101
  - 01                   0110101110
  - 111000                000111
- Every Binary number has a corresponding
- Ding Decimal Value (and vice versa)
- examples:

Binary Number    Decimal Equivalent

1	1
10	2
11	3
...	...
1010111	87

- Each position for a binary number has a value.
- Add up all of the products to get the final result .
- In general , the “position values” in a binary number are the powers of two.
  - The first position value is 2<sup>0</sup>, ie, one.
  - The 2nd position value is 2<sup>1</sup>, ie, two.
  - The 2nd position value is 2<sup>2</sup>, ie, four.
  - The 2nd position value is 2<sup>3</sup>, ie, eight.

→ The 2nd position values is 24, ie sixteen.

## Example

- The value of binary 01101001 is decimal 105. This is worked out below:

128	64	32	16	8	4	2	1	
<hr/>								
0	1	1	0	1	0	0	1	
							1 x 1	= 1
								= 0
					0 x 4			= 0
				1 x 8				= 8
			0 x 16					= 0
		1 x 32						= 32
	1 x 64							= 64
0 x 128								= 0
<hr/>								
								Answer: 105

## Another example

- The value of binary 10011100 is decimal 156. This is worked out below:

[illegible]

## Bitwise Operators (tutorial point)

Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) = 12, i.e., 0000 1100
	Binary OR Operator copies a bit if it exists in either operand.	(A   B) = 61, i.e., 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) = 49, i.e., 0011 0001
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) = -61, i.e., 1100 0011 in 2's complement form.
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 = 240 i.e., 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 = 15 i.e., 0000 1111

{ } Learn Programming

## Truth Table

p	q	p & q	p   q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

6 3 1 2 4 5