PASCAL - BIT OPERATORS

The Bitwise operators supported by Pascal are listed in the following table. Assume variable A holds 60 and variable B holds 13, then −

Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	A & B will give 12, which is 0000 1100
|	Binary OR Operator copies a bit if it exists in either operand.	$A \mid B$ will give 61, which is 0011 1101
!	Binary OR Operator copies a bit if it exists in either operand. Its same as | operator.	$A ! B$ will give 61, which is 0011 1101
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	A will give -61, which is 1100 0011 in 2's complement form due to a signed binary number.
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240, which is 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 will give 15, which is 0000 1111

Please note that different implementations of Pascal differ in bitwise operators. Free Pascal, the compiler we used here, however, supports the following bitwise operators −

Operators	Operations
not	Bitwise NOT
and	Bitwise AND
or	Bitwise OR
xor	Bitwise exclusive OR
shl	Bitwise shift left
shr	Bitwise shift right
<<	Bitwise shift left
>>	Bitwise shift right

The following example illustrates the concept −

```
program beBitwise;
var
a, b, c: integer;

begin
   a := 60; (* 60 = 0011 1100 *)
   b := 13; (* 13 = 0000 1101 *)
```

```
   c := 0;

   c := a and b;       (* 12 = 0000 1100 *)
   writeln('Line 1 - Value of c is  ', c );

   c := a or b;       (* 61 = 0011 1101 *)
   writeln('Line 2 - Value of c is  ', c );

   c := not a;          (* -61 = 1100 0011 *)
   writeln('Line 3 - Value of c is  ', c );

   c := a << 2;     (* 240 = 1111 0000 *)
   writeln('Line 4 - Value of c is  ', c );

   c := a >> 2;     (* 15 = 0000 1111 *)
   writeln('Line 5 - Value of c is  ', c );
end.
```

When the above code is compiled and executed, it produces the following result −

```
Line 1 - Value of c is 12
Line 2 - Value of c is 61
Line 3 - Value of c is -61
Line 4 - Value of c is 240
Line 5 - Value of c is 15
```