Chapter -3: Introduction to Python Language

Logical Operators

Logical operators are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Example #1:

```
# Python program to demonstrate
# logical and operator
a = 10
b = 10
c = -10

if a > 0 and b > 0:
   print("The numbers are greater than 0")

if a > 0 and b > 0 and c > 0:
   print("The numbers are greater than 0")
else:
   print("Atleast one number is not greater than 0")
```

Output:

```
The numbers are greater than 0
Atleast one number is not greater than 0
```

Example #2:

```
# Python program to demonstrate
# logical and operator

a = 10
b = 12
c = 0

if a and b and c:
   print("All the numbers have boolean value as True")
else:
   print("Atleast one number has boolean value as False")
```

Output:

```
Atleast one number has boolean value as False
```

Example #3:

```
# Python program to demonstrate
# logical not operator

a = 10

if not a:
   print("Boolean value of a is True")

if not (a%3 == 0 or a%5 == 0):
   print("10 is not divisible by either 3 or 5")
else:
   print("10 is divisible by either 3 or 5")
```

Output:

```
10 is divisible by either 3 or 5
```

Identity Operators

Identity operators are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Example

```
a = 20
b = 20

if ( a is b ):
   print ("Line 1 - a and b have same identity")
else:
   print ("Line 1 - a and b do not have same identity")

if ( id(a) == id(b) ):
   print ("Line 2 - a and b have same identity")
else:
   print ("Line 2 - a and b do not have same identity")

b = 30
if ( a is b ):
   print ("Line 3 - a and b have same identity")
else:
   print ("Line 3 - a and b do not have same identity")

if ( a is not b ):
   print ("Line 4 - a and b do not have same identity")
else:
   print ("Line 4 - a and b have same identity")
```

When you execute the above program it produces the following result −

Line 1 - a and b have same identity
Line 2 - a and b have same identity
Line 3 - a and b do not have same identity
Line 4 - a and b do not have same identity

Membership Operators

Membership operators are used to test if a sequence is presented in an object:

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

Example

```
a = 10
b = 20
list = [1, 2, 3, 4, 5 ];

if ( a in list ):
   print ("Line 1 - a is available in the given list")
else:
   print ("Line 1 - a is not available in the given list")

if ( b not in list ):
   print ("Line 2 - b is not available in the given list")
else:
   print ("Line 2 - b is available in the given list")

a = 2
if ( a in list ):
   print ("Line 3 - a is available in the given list")
else:
   print ("Line 3 - a is not available in the given list")
```

When you execute the above program it produces the following result −

Line 1 - a is not available in the given list

Line 2 - b is not available in the given list

Line 3 - a is available in the given list

Bitwise Operators

Bitwise operators are used to compare (binary) numbers:

Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

Example

```
a = 60            # 60 = 0011 1100
b = 13            # 13 = 0000 1101
c = 0

c = a & b;        # 12 = 0000 1100
print ("Line 1 - Value of c is ", c)

c = a | b;        # 61 = 0011 1101
print ("Line 2 - Value of c is ", c)

c = a ^ b;        # 49 = 0011 0001
print ("Line 3 - Value of c is ", c)

c = ~a;           # -61 = 1100 0011
print ("Line 4 - Value of c is ", c)

c = a << 2;       # 240 = 1111 0000
print ("Line 5 - Value of c is ", c)

c = a >> 2;       # 15 = 0000 1111
print ("Line 6 - Value of c is ", c)
```

When you execute the above program 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 49
Line 4 - Value of c is -61
Line 5 - Value of c is 240
Line 6 - Value of c is 15
```