BCA – 502: Python Programming

Rahul Kumar Singh

In today's Class we have discussed on Comparison (Relational) Operators, Logical Operators, and Membership Operators.

Python Comparison Operators:-

These operators compare the values on either sides of them and decide the relation among them. They are also called Relational operators. Comparison operators are used to compare two values:

Some Comparison Operators are as below:

Equal (==) Operator:- If the values of two operands are equal, then the condition becomes true.

Example:-

```
x = 5

y = 3

print(x == y)

# returns False because 5 is not equal to 3
```

Output:- False

Not Equal (!=) Operator:- If values of two operands are not equal, then condition becomes true.

Example:-

```
x = 5

y = 3

print(x != y)

# returns True because 5 is not equal to 3
```

Output:- True

Greater than (>) operator:- If the value of left operand is greater than the value of right operand, then condition becomes true.

Example:-

```
x = 5

y = 3

print(x > y)

# returns True because 5 is greater than 3
```

Output:- True

Less than (<) operator:- If the value of left operand is less than the value of right operand, then condition becomes true.

Example:-

```
x = 5

y = 3

print(x < y)

# returns False because 5 is not less than 3
```

Output:- False

Greater than or equal to (>=) operator:- If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.

Example:-

```
x = 5

y = 3

print(x >= y)

# returns True because five is greater, or equal, to 3
```

Output:- True

Less than or equal to (<=) operator:- If the value of left operand is less than or equal to the value of right operand, then condition becomes true.

Example:-

```
x = 5

y = 3
```

```
print(x <= y)

# returns False because 5 is neither less than or equal to 3
```

Output:- False

Q.) Write a program in Python to perform all Comparison operation.

Ans:-

```
#!/usr/bin/python

a = 21

b = 10

c = 0


if ( a == b ):
   print "Line 1 - a is equal to b"
else:
   print "Line 1 - a is not equal to b"


if ( a != b ):
   print "Line 2 - a is not equal to b"
```

```
else:
   print "Line 2 - a is equal to b"

if ( a <> b ):
   print "Line 3 - a is not equal to b"
else:
   print "Line 3 - a is equal to b"

if ( a < b ):
   print "Line 4 - a is less than b"
else:
   print "Line 4 - a is not less than b"

if ( a > b ):
   print "Line 5 - a is greater than b"
else:
   print "Line 5 - a is not greater than b"

a = 5;
b = 20;
```

```
if ( a <= b ):
   print "Line 6 - a is either less than or equal to  b"
else:
   print "Line 6 - a is neither less than nor equal to  b"

if ( b >= a ):
   print "Line 7 - b is either greater than  or equal to b"
else:
   print "Line 7 - b is neither greater than  nor equal to b"
```

Output:- Line 1 - a is not equal to b

Line 2 - a is not equal to b

Line 3 - a is not equal to b

Line 4 - a is not less than b

Line 5 - a is greater than b

Line 6 - a is either less than or equal to b

Line 7 - b is either greater than or equal to b

Python Logical Operators:-

Logical operators are used to combine conditional statements. There are following logical operators supported by Python language.

and (Logical AND) operator:- If both the operands are true then condition becomes true. It returns True if both statements are true.

Example:-

```
x = 5

print(x > 3 and x < 10)

# returns True because 5 is greater than 3 AND 5 is less than 10
```

Output:- True

or (Logical OR) operator:- If any of the two operands are non-zero then condition becomes true. It returns True if one of the statements is true.

Example:-

```
x = 5

print(x > 3 or x < 4)

# returns True because one of the conditions are true (5 is greater than 3, but 5 is not less than 4)
```

Output:- True

not (Logical NOT) operator:- Used to reverse the logical state of its operand.Reverse the result, returns

False if the result is true.

Example

```
x = 5

print(not(x > 3 and x < 10))

# returns False because not is used to reverse the result
```

Output:- False

Python Membership Operators:-

Python's membership operators test for membership in a sequence, such as strings, lists, or tuples. Membership operators are used to test if a sequence is presented in an object. There are two membership operators as explained below −

in operator :- Returns True if a sequence with the specified value is present in the object. Evaluates to true if it finds a variable in the specified sequence and false otherwise.

Example:-

```
x = ["apple", "banana"]

print("banana" in x)

# returns True because a sequence with the value "banana" is in the list
```

Output:- True

not in operator:- Evaluates to true if it does not finds a variable in the specified sequence and false otherwise.not in Returns True if a sequence with the specified value is not present in the object

Example:-

```
x = ["apple", "banana"]

print("pineapple" not in x)

# returns True because a sequence with the value "pineapple" is not in the list
```

Output:- True

Q.) Write a Python programme to to perform Membership Operators.

```
#!/usr/bin/python

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"
```

Output:-

```
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
```