Lecture#9 Python Programming

What is Python Not equal to Operator?

Python is identified as a programming language that is very dynamic, and it is generally regarded as a strongly typed language. This statement can be explained by understanding the significance of not equal operator. In not equal operator, if the values of the two operands on either side of the operator are not equal, then the operator provides true value, else it provides false.

In not equal operator, if two variables are of different types but hold the same values in themselves, then the not equal operator returns a true. Not many programming languages can classify it as true if the variable type is of a different type, which makes python a very dynamic language. In python, not equal operators can be classified as one of the comparison operators.

In this Python tutorial, you will learn,

- **Types of Not equal operators and Syntax in Python**
- **A simple example of not equal operator**
- **How to use Not Equal Operator with IF Statement**
- **How to use equal to (==) operator with while loop**
- **An example of getting even numbers by using not equal operator**
- **How to use Python not equal Operator with custom object**
- **Comparison operators in Python**
- **Useful Tips on Usage of Not Equal Operator**

Types of Not equal to operators with Syntax in Python

There are two types of not equal operators in python:-

!=

<>

The first type, “!=” is used in python versions 2 and 3.

The second type, “<>,” is used in python version 2, and under version 3, this operator is deprecated.

The syntax of both types is shown below: –

```
X<>Y
X!=Y
```

A simple example of not equal operator

Let us consider two scenarios to illustrate not equal operator. Following is the example of not equal operator for same data type but different values:-

```
A = 44
B = 284
C = 284
print(B!=A)
print(B!=C)
```

Output:

```
True
False
```

Following is the example of not equal operator for different data type but same values

```
C = 12222
X = 12222.0
Y = "12222"
print(C!=X)
print(X!=Y)
print(C!=Y)
```

Output:

```
False
True
True
```

How to use Not Equal Operator with IF Statement

In python, an if-statement can be described as a statement that checks the entry-level condition and executes when it is true.

Let us take a basic example of using if statement and not equal to operator as shown below: –

```
X = 5
Y = 5
if ( X != Y ):
  print("X is not equal to Y")
else:
  print("X is equal to Y")
```

Output:

```
X is equal to Y
```

Here, not equal to (!=) is utilized along with the if statement.

How to use equal to (==) operator with while loop

In python, while-loop iterates block of code as long as a condition is true or false. Let us take a case of printing odd numbers using while loop and equal to operator as shown below: –

```
m = 300
while m <= 305:
   m = m + 1
  if m%2 == 0:
     continue
   print (m)
```

Output:

```
301
303
305
```

Here, equal to (==) is utilized along with the if statement.

Example: Finding even numbers by using not equal operator

In python, while loop can also be used with not equal to operator. Let us take a case of printing even numbers using while loop and not equal to operator as shown below: –

```
m = 300
while m <= 305:
  m = m + 1
  if m%2 != 0:
    continue
  print (m)
```

Output:

```
302
304
306
```

Here, not equal to (!=) is utilized along with the if statement.

Comparison operators in Python

Following table describes the list of comparison operators in python: –

Operator	Meaning	Example
!=	Not equal to-gives true if operands do not have the same values	A!=B
==	Equal to-Gives true if operands have the same values	A==B
>=	Greater than or equal to- gives true as value if the first operand is greater than or equal with the second operand	A>=B
<=	Less than or equal to- gives true as value if the first operand is Less than or equal with the second operand	A<=B
>	Greater than – gives true as value if the first operand is greater than the second operand	A>B
<	Less than – gives true as value if the first operand is Less than the second operand	A<B