What is Operator in python and Types of operators

In Python, operators are special symbols or keywords that perform operations on operands (variables or values). Python supports various types of operators, which can be classified into several categories:

We have several types of operators in python. They are,

- ❖ Arithmetic Operators
- ❖ Relational / Comparison Operators
- ❖ Logical Operators
- ❖ Assignment Operators
- ❖ Membership Operators
- ❖ Identity Operators
- ❖ Ternary operator

Arithmetic Operators:

Arithmetic operators perform mathematical operations like addition, subtraction, multiplication, division, etc.

Arithmetic operators returns **Number** type result like **int, float**

- Addition: **+**
- Subtraction: **-**
- Multiplication: *****
- Division: **/**
- Modulus (Remainder): **%**
- Exponentiation: ******
- Floor Division (Integer Division): **//**

For Example:

```
a = 10
b = 3

print(a + b)        # Output: 13
print(a - b)        # Output: 7
print(a * b)        # Output: 30
print(a / b)        # Output: 3.3333333333333335
print(a % b)        # Output: 1
print(a ** b)       # Output: 1000
print(a // b)       # Output: 3
```

Comparison Operators:

Comparison operators are used to compare values. They return **True** or **False** as a result.

- Equal to: **==**
- Not equal to: **!=**
- Greater than: **>**
- Less than: **<**
- Greater than or equal to: **>=**
- Less than or equal to: **<=**

For example:

```
x = 5
y = 10
print(x == y)     # Output: False
print(x != y)     # Output: True
print(x > y)        # Output: False
print(x < y)        # Output: True
print(x >= y)     # Output: False
print(x <= y)     # Output: True
```

Logical Operators:

Logical operators are used to combine conditional statements. Returns **True** or **False** values.

- Logical AND: **and**
- Logical OR: **or**
- Logical NOT: **not**

and truth table

condition1	condition2	Result
True	True	True
True	False	False
False	True	False
False	False	False

or truth table

condition1	condition2	Result
True	True	True
True	False	True
False	True	True
False	False	False

Note: **not** operator returns **True** value as **False** and **False** value as **True**

For example

```
a = True
b = False
print(a and b)        # Output: False
print(a or b)         # Output: True
print(not a)          # Output: False
```

Assignment Operators:

Assignment operators are used to assign values to variables.

- Assignment: **=**
- Add and assign: **+=**
- Subtract and assign: **-=**
- Multiply and assign: ***=**
- Divide and assign: **/=**
- Modulus and assign: **%=**
- Exponentiation and assign: ****=**
- Floor division and assign: **//=**

For Example:

```
x = 5
x += 2        # Equivalent to:    x = x + 2
print(x)      # Output: 7
```

Membership Operators:

Membership operators are used to check if a value or variable is found in a sequence.

- **in**: Returns True if a value is found in the specified sequence.
- **not in**: Returns True if a value is not found in the specified sequence.

For example:

```
my_list = [1, 2, 3, 4, 5]

print(3 in my_list)           # Output: True

print(6 not in my_list)       # Output: True
```

Identity Operators:

Identity operators are used to compare the memory locations of two objects.

- **is**: Returns True if both variables point to the same object.
- **is not**: Returns True if both variables point to different objects.

For example:

```
x = [1, 2, 3]
y = [1, 2, 3]
z = x

print(x is z)        # Output: True
print(x is y)        # Output: False
print(x is not y)    # Output: True
```

Ternary Operator:

Python supports the ternary operator, also known as the conditional expression. The syntax of the ternary operator in Python is:

x if condition else y

This expression evaluates to **x** if the condition is true, and **y** otherwise.

Here's an example demonstrating the use of the ternary operator in Python:

```
# Assigning a value based on a condition
x = 10
y = 20
result = x if x > y else y
print(result)  # Output: 20

# Using the ternary operator in a print statement
print("x is greater than y" if x > y else "y is greater than or equal to x")
# Output: "y is greater than or equal to x"
```

In the first example, the value of **result** is assigned **x** if **x** is greater than **y**, otherwise, it's assigned **y**.

In the second example, the ternary operator is used directly within the **print** statement to determine which message to print based on the comparison between **x** and **y**.

The ternary operator provides a concise and readable way to express conditional expressions in Python.

These are the main types of operators in Python, each serving different purposes and used in various contexts within Python programming. Understanding them is crucial for effective programming and problem-solving.