AMRITA VISHWA VIDYAPEETHAM

Python Operators

Anoop S Babu
Faculty Associate
Dept. of Computer Science & Engineering
bsanoop@am.amrita.edu

Python Operators

- Operators are special symbols used to **perform operations** (arithmetic or logical) in **values** and **variables**.
- The value that the operator operates on is called the **operand**.
- Python divides the operators in the following groups:
  - Arithmetic operators
  - Comparison operators
  - Logical operators
  - Bitwise operators
  - Assignment operators
  - Identity operators
  - Membership operators

Arithmetic Operators

Operator	Meaning	Example
+	Add two operands or unary plus	x + y+ 2
-	Subtract right operand from the left or unary minus	x - y- 2
*	Multiply two operands	x * y
/	Divide left operand by the right one (always results into float)	x / y
%	Modulus - remainder of the division of left operand by the right	x % y (remainder of x/y)
//	Floor division - division that results into whole number adjusted to the left in the number line	x // y
**	Exponent - left operand raised to the power of right	x**y (x to the power y)

Example 1: Arithmetic operators in Python

```
x = 15
y = 4

print('x + y =',x+y)

print('x - y =',x-y)

print('x * y =',x*y)

print('x / y =',x/y)

print('x // y =',x//y)

print('x ** y =',x**y)
```

Output

```
x + y = 19
x - y = 11
x * y = 60
x / y = 3.75
x // y = 3
x ** y = 50625
```

Comparison Operators

- Comparison operators are used to **compare values**. It returns either **True** or **False** according to the condition.

Operator	Meaning	Example
>	Greater than - True if left operand is greater than the right	x > y
<	Less than - True if left operand is less than the right	x < y
==	Equal to - True if both operands are equal	x == y
!=	Not equal to - True if operands are not equal	x != y
>=	Greater than or equal to - True if left operand is greater than or equal to the right	x >= y
<=	Less than or equal to - True if left operand is less than or equal to the right	x <= y

Example 2: Comparison operators in Python

```
x = 10
y = 12

print('x > y is',x>y)

print('x < y is',x<y)

print('x == y is',x==y)

print('x != y is',x!=y)

print('x >= y is',x>=y)

print('x <= y is',x<=y)
```

Output

```
x > y is False
x < y is True
x == y is False
x != y is True
x >= y is False
x <= y is True
```

Logical Operators

Operator	Meaning	Example
and	True if both the operands are true	x and y
or	True if either of the operands is true	x or y
not	True if operand is false (complements the operand)	not x

Example 3: Logical Operators in Python

```
x = True
y = False

print('x and y is',x and y)

print('x or y is',x or y)

print('not x is',not x)
```

Output

```
x and y is False
x or y is True
not x is False
```

Bitwise Operators

- Bitwise operators act on operands as if they were strings of binary digits. They **operate bit by bit**.
- **In the table below:** Let x = 10 (**0000 1010** in binary) and y = 4 (**0000 0100** in binary)

Operator	Meaning	Example
&	Bitwise AND	x & y = 0 (0000 0000)
|	Bitwise OR	x | y = 14 (0000 1110)
~	Bitwise NOT	~x = -11 (1111 0101)
^	Bitwise XOR	x ^ y = 14 (0000 1110)
>>	Bitwise right shift	x >> 2 = 2 (0000 0010)
<<	Bitwise left shift	x << 2 = 40 (0010 1000)

Assignment Operators

- Assignment operators are used to **assign values to variables.**

Operator	Example	Equivalent to
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=	x /= 5	x = x / 5
%=	x %= 5	x = x % 5
//=	x //= 5	x = x // 5
**=	x **= 5	x = x ** 5
&=	x &= 5	x = x & 5
|=	x |= 5	x = x | 5
^=	x ^= 5	x = x ^ 5
>>=	x >>= 5	x = x >> 5
<<=	x <<= 5	x = x << 5

Identity Operators

- Used to check if two values (or variables) are **located on the same part of the memory**.
- Two variables that are equal **does not imply** that they are identical.

Operator	Meaning	Example
is	True if the operands are identical (refer to the same object)	x is True
is not	True if the operands are not identical (do not refer to the same object)	x is not True

Example 4: Identity operators in Python

```
x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1,2,3]
y3 = [1,2,3]

print(x1 is not y1)

print(x2 is y2)

print(x3 is y3)
```

Output

```
False
True
False
```

Membership Operators

- Used to test whether a **value** or **variable** is **found in a sequence** (string, list, tuple, set and dictionary).

Operator	Meaning	Example
in	True if value/variable is found in the sequence	5 in x
not in	True if value/variable is not found in the sequence	5 not in x