Operators in Python

Operators are special symbols in Python that perform operations on variables and values. Python supports a variety of operators, which are categorized as follows:

Arithmetic Operators

Arithmetic operators are used to perform mathematical operations.

Operator	Description	Example
+	Addition	a + b
-	Subtraction	a - b
*	Multiplication	a * b
/	Division	a / b
%	Modulus	a % b
**	Exponentiation	a ** b
//	Floor Division	a // b

```
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 two values.

Operator	Description	Example
==	Equal to	a == b
!=	Not equal to	a != b
>	Greater than	a > b
<	Less than	a < b
>=	Greater than or equal to	a >= b
<=	Less than or equal to	a <= b

```
a = 10
b = 3
```

```
print(a == b)  # Output: False
print(a != b)  # Output: True
print(a > b)   # Output: True
print(a < b)   # Output: False
print(a >= b)  # Output: True
print(a <= b)  # Output: False
```

Logical Operators

Logical operators are used to combine conditional statements.

Operator	Description	Example
and	Logical AND	a and b
or	Logical OR	a or b
not	Logical NOT	not a

```
a = True
b = False

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

Bitwise Operators

Bitwise operators are used to perform bit-level operations.

Operator	Description	Example
&	Bitwise AND	a & b
|	Bitwise OR	a | b
ˆ	Bitwise XOR	a ˆ b
~	Bitwise NOT	~a
«	Bitwise left shift	a « b
»	Bitwise right shift	a » b

```
a = 10  # 1010 in binary
b = 4   # 0100 in binary

print(a & b)  # Output: 0
print(a | b)  # Output: 14
print(a ^ b)  # Output: 14
print(~a)     # Output: -11
print(a << 1) # Output: 20
print(a >> 1) # Output: 5
```

Assignment Operators

Assignment operators are used to assign values to variables.

Operator	Description	Example
=	Assign	a = 5
+=	Add and assign	a += 5
-=	Subtract and assign	a -= 5
*=	Multiply and assign	a *= 5
/=	Divide and assign	a /= 5
%=	Modulus and assign	a %= 5
//=	Floor divide and assign	a //= 5
**=	Exponentiate and assign	a **= 5
&=	Bitwise AND and assign	a &= 5
|=	Bitwise OR and assign	a |= 5
ˆ=	Bitwise XOR and assign	a ˆ= 5
»=	Bitwise right shift and assign	a »= 5
«=	Bitwise left shift and assign	a «= 5

```
a = 10
b = 3

a += b  # Equivalent to a = a + b
print(a)  # Output: 13

a -= b  # Equivalent to a = a - b
print(a)  # Output: 10

a *= b  # Equivalent to a = a * b
print(a)  # Output: 30

a /= b  # Equivalent to a = a / b
print(a)  # Output: 10.0

a %= b  # Equivalent to a = a % b
print(a)  # Output: 1.0

a //= b  # Equivalent to a = a // b
print(a)  # Output: 0.0

a **= b  # Equivalent to a = a ** b
print(a)  # Output: 0.0

a &= b  # Equivalent to a = a & b
print(a)  # Output: 0.0
```

```python
a |= b  # Equivalent to a = a | b
print(a)  # Output: 0.0

a ^= b  # Equivalent to a = a ^ b
print(a)  # Output: 0.0

a = 10
a >>= b  # Equivalent to a = a >> b
print(a)  # Output: 1

a <<= b  # Equivalent to a = a << b
print(a)  # Output: 8
```

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	a in b
not in	Returns True if a sequence with the specified value is not present in the object	a not in b

```python
a = [1, 2, 3, 4, 5]
print(3 in a)     # Output: True
print(6 not in a) # Output: True
```

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	a is b
is not	Returns True if both variables are not the same object	a is not b

```python
a = [1, 2, 3]
b = [1, 2, 3]
c = a
```

```
print(a is b)     # Output: False
print(a is c)     # Output: True
print(a is not b) # Output: True
```

Stay Updated

Be sure to this repository to stay updated with new examples and enhancements!

License

Contact

Panagiotis Moschos - pan.moschos86@gmail.com

Note: This is a Python script and requires a Python interpreter to run.

Happy Coding

Made with by Panagiotis Moschos (https://github.com/pmoschos)