Operators in Python?

Operators are **symbols** or **words** used to perform **operations** on variables and values.

1. **Arithmetic Operators**
2. **Assignment Operators**
3. **Comparison Operators**
4. **Logical Operators**
5. **Identity Operators**
6. **Membership Operators**
7. **Bitwise Operators**

1 Arithmetic Operators

Used to perform **basic math operations**.

Operator	Description	Example	Output
+	Addition	5 + 3	8
-	Subtraction	5 - 3	2
*	Multiplication	5 * 3	15
/	Division	5 / 2	2.5
//	Floor Division	5 // 2	2
%	Modulus (Remainder)	5 % 2	1
**	Exponentiation	2 ** 3	8

---

2 Assignment Operators

Used to assign values to variables.

Operator	Description	Example	Same As
=	Assign	x = 5	—
+=	Add and assign	x += 2	x = x + 2
-=	Subtract and assign	x -= 2	x = x - 2
*=	Multiply and assign	x *= 3	x = x * 3
/=	Divide and assign	x /= 2	x = x / 2
//=	Floor divide and assign	x //= 2	x = x // 2
%=	Modulus and assign	x %= 2	x = x % 2
**=	Power and assign	x **= 2	x = x ** 2

---

3 Comparison Operators

Used to **compare** two values. Returns **True or False**.

Operator	Description	Example
`==`	Equal to	`5 == 5` → `True`
`!=`	Not equal to	`5 != 3` → `True`
`>`	Greater than	`5 > 3` → `True`
`<`	Less than	`5 < 3` → `False`
`>=`	Greater than or equal	`5 >= 5` → `True`
`<=`	Less than or equal	`5 <= 3` → `False`

---

4 Logical Operators

Used to combine **Boolean** expressions.

Operator	Description	Example	Result
`and`	Both true	`True and False`	`False`
`or`	At least one true	`True or False`	`True`
`not`	Inverts result	`not True`	`False`

---

5 Identity Operators

Used to compare **memory locations** of two objects.

Operator	Description	Example	Result
`is`	Same object	`x is y`	`True` if same object
`is not`	Not same object	`x is not y`	`True` if different

Example:

```
a = [1, 2]
b = a
c = [1, 2]
print(a is b)      # True
print(a is c)      # False (same value, different object)
```

6️⃣ Membership Operators

Used to test if a value is a member of a sequence.

Operator	Description	Example	Result
in	Value is in the sequence	"a" in "apple"	True
not in	Value not in sequence	"z" not in "apple"	True

7️⃣ Bitwise Operators

Works on bits (0s and 1s). More advanced, used in low-level programming.

Operator	Name	Description	Example	Result (in binary)	Result (in decimal)
&	AND	Sets each bit to 1 **if both** bits are 1	5 & 3	0101 & 0011 = 0001	1
|	OR	Sets each bit to 1 **if one** of two bits is 1	5 | 3	0101 | 0011 = 0111	7
^	XOR	Sets each bit to 1 **if only one** of two bits is 1	5 ^ 3	0101 ^ 0011 = 0110	6
~	NOT	Inverts all the bits (1 becomes 0, and 0 becomes 1) and adds -1	~5	~00000101 = 11111010	-6 (in 2's complement)
<<	Left Shift	Shifts bits to the left, adds 0 at right	5 << 1	00000101 << 1 = 00001010	10
>>	Right Shift	Shifts bits to the right, drops bits at right	5 >> 1	00000101 >> 1 = 00000010	2