

03 - Operators



Operators

- Special symbols or keywords used to perform operations on variables and values.

Operators

- Special symbols or keywords used to perform operations on variables and values.
- Operations can act on one or more operands.

Operators

- Special symbols or keywords used to perform operations on variables and values.
- Operations can act on one or more operands.
- If the operation involves a single operand, then the operator is unary. If the operator involves two operands, then the operator is binary.

Operators

Special symbols or keywords used to perform operations on variables and values.

Python divides the operators into **seven categories**

1. Arithmetic operators
2. Assignment operators
3. Relational operators
4. Logical operators
5. Bitwise operators
6. Identity operators
7. Membership operators

```
number_of_operators = 7

print("How many operators does Python have?")

print("Answer: ", number_of_operators)
```

Arithmetic Operators

```
number_of_operators = 7

print("How many operators does Python have?")

print("Answer: ", number_of_operators)
```

Arithmetic Operators

Arithmetic operators are used with numeric values to perform common mathematical operators. Such as:

- Addition
- Subtraction
- Multiplication
- Division
- Floor Division
- Modulus
- Exponentiation

```
number_of_operators = 7

print("How many operators does Python have?")

print("Answer: ", number_of_operators)
```

Arithmetic Operators - Cont'd

Add

```
x = 10
y = 20

print(x + y)    # 30

print(x + 30)   # 40

print(40 + 50)  # 90
```

Subtract

```
x = 10
y = 20

print(x - y)    # -10

print(x - 30)   # -20

print(40 - 50)  # -10
```

Multiply

```
x = 10
y = 20

print(x * y)    # 200

print(x * 30)   # 300

print(40 * 50)  # 2000
```

Divide

```
x = 10
y = 20

print(x / y)     # 5.0

print(x / 30)    # 0.333

print(40 / 50)   # 0.8
```


Assignment Operators



Assignment Operators

Assignment operators are used to assign values to variables.

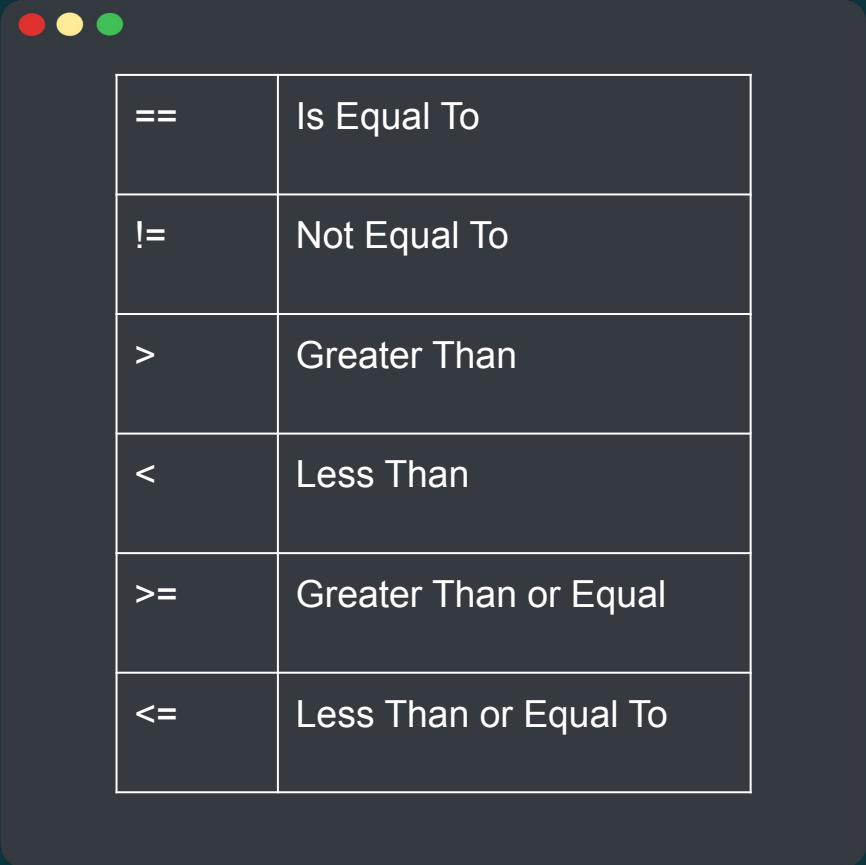
=	Assignment operator
+=	Addition assignment
-=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
//=	Floor division assignment
%=	Remainder assignment
**=	Exponent assignment

Relational Operators



Relational Operators

Relational operators are used to compare between two values and return a Boolean result (**True** or **False**).



==	Is Equal To
!=	Not Equal To
>	Greater Than
<	Less Than
>=	Greater Than or Equal
<=	Less Than or Equal To

Logical Operators



Logical Operators

Logical operators are used to combine multiple conditions and evaluate them to True or False.

a and b	True if both the operands are True
a or b	True if at least one of the operands is True
not a	True if the operand is False and vice-versa

Bitwise Operators



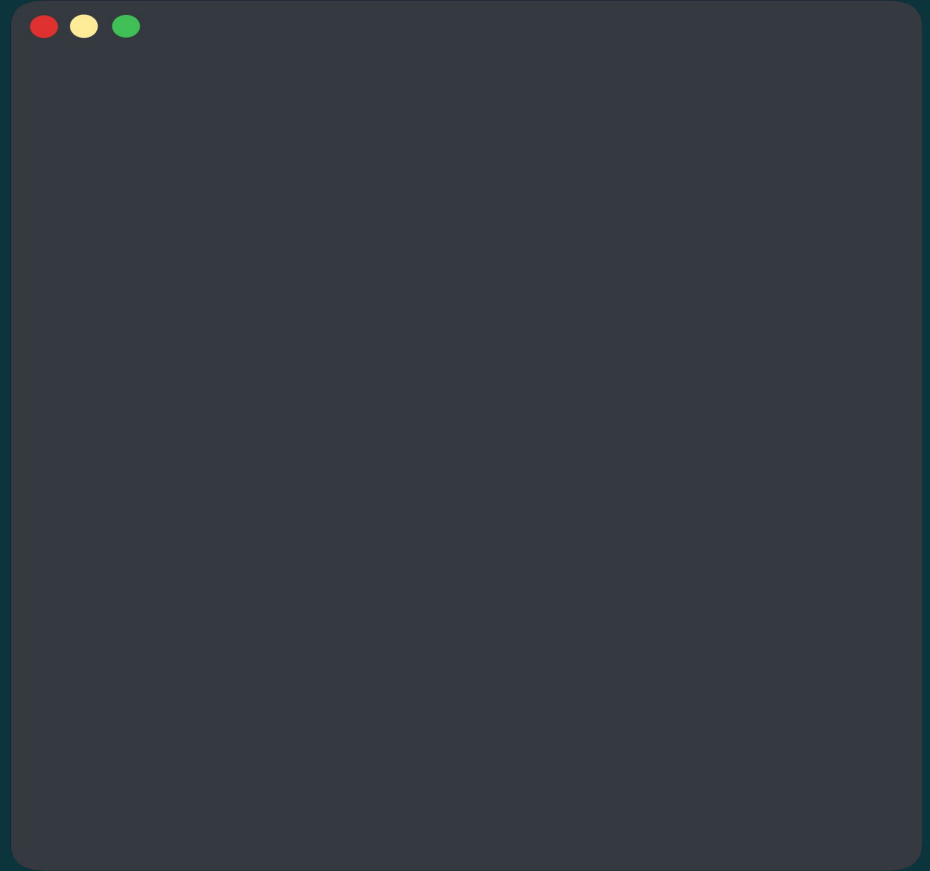
Bitwise Operators

Bitwise operators act on operands at the **bit level**.

Bitwise operators are often used in systems programming or working with hardware for tasks like setting, clearing, or toggling bits in a number.

&	Bitwise AND
	Bitwise OR
~	Bitwise NOT
^	Bitwise XOR
>>	Bitwise right shift
<<	Bitwise left shift

Identity Operators

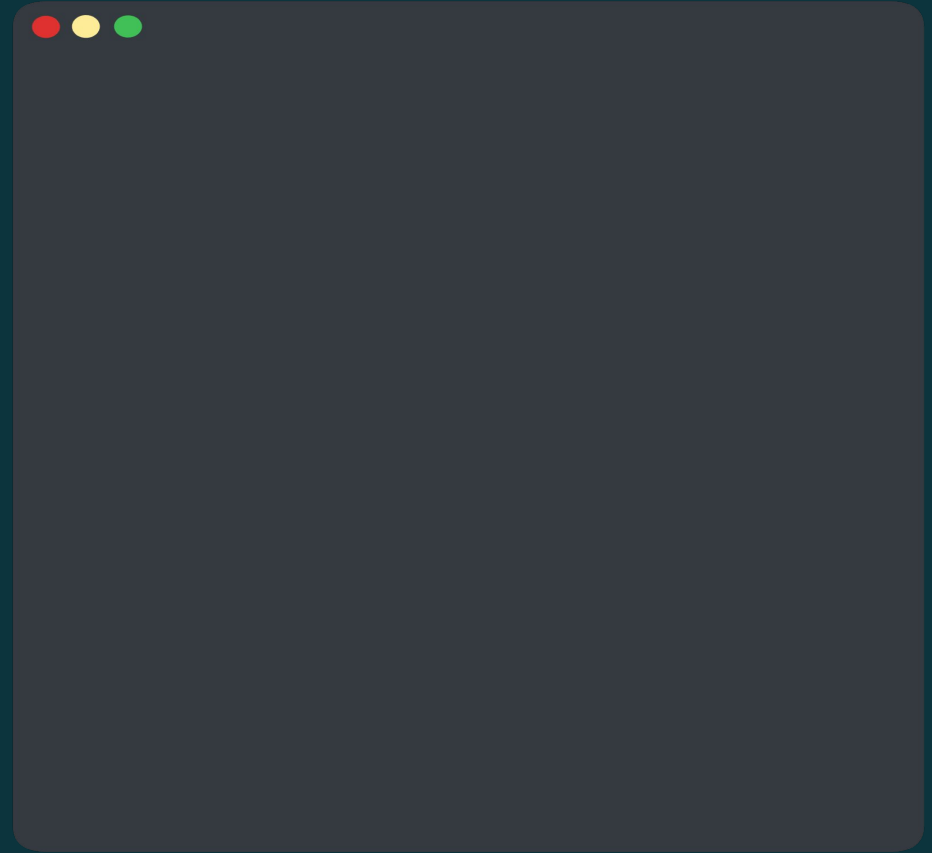


Identity Operators

Identity operators are used to check if two values (objects) refer to the same memory location.

is	True if the operands are identical (refer to the same object)
is not	True if the operands are not identical (do not refer to the same object)

Membership Operators



Membership Operators

Membership operators are used to check if a value exists in a sequence

(e.g. string, list, tuple, set, dictionary)

in	True if value/variable is found in the sequence
not in	True if value/variable is not found in the sequence

Conclusion

Types of Operators	
Arithmetic	+, -, *, /, //, %, **
Relational	==, !=, >=, >, <=, <
Logical	and, or, not
Bitwise	&, , ~, ^, <<, >>
Assignment	=, +=, -=, *=, /=, //=, %=, **=, &=, =, ^=, <<=, >>=
Identity	is, is not
Membership	in, not in