

Type Conversion

- Python allows converting between data types using built-in functions such as `int()`, `float()`, `str()`, `list()`, `tuple()`, etc.:

python

```
x = "100"  
y = int(x) # Converts string to integer
```

What are Operators?

Operators are special symbols or keywords that perform specific operations on variables, values, or expressions. They are fundamental in writing logic and performing calculations in Python programs.

Types of Operators in Python

1. Arithmetic Operators

Used for basic mathematical operations.

Operator	Operation	Example	Result
+	Addition	5 + 3	8
-	Subtraction	5 - 3	2
*	Multiplication	5 * 3	15
/	Division	5 / 3	1.6667
//	Floor Division	5 // 3	1
%	Modulus (Remainder)	5 % 3	2
**	Exponentiation	5 ** 3	125

Example:

python

```
a = 10
b = 3
print(a + b) # 13
print(a ** b) # 1000
```

2. Assignment Operators

Used to assign values to variables.

Operator	Example	Description
=	a = 5	Simple assignment
+=	a += 3 (a = a + 3)	Add and assign
-=	a -= 2 (a = a - 2)	Subtract and assign
*=	a *= 4 (a = a * 4)	Multiply and assign
/=	a /= 2 (a = a / 2)	Divide and assign

3. Comparison (Relational) Operators

Used to compare two values and return a Boolean result (True or False).

Operator	Meaning	Example	Result
==	Equal to	5 == 3	False
!=	Not equal to	5 != 3	True
>	Greater than	5 > 3	True
<	Less than	5 < 3	False
>=	Greater or equal	5 >= 5	True
<=	Less or equal	5 <= 3	False

4. Logical Operators

Used to combine conditional statements.

Operator	Meaning	Example
and	True if both true	True and False → False
or	True if one true	True or False → True
not	Inverts boolean	not True → False

Example:

python

```
x = 5
print(x > 3 and x < 10) # True
print(not(x > 3))      # False
```

5. Bitwise Operators

Operate on bits of integers.

Operator	Description	Example
&	Bitwise AND	5 & 3 = 1
	Bitwise OR	5 3 = 7
^	Bitwise XOR	5 ^ 3 = 6
~	Bitwise NOT	~5 = -6
<<	Left shift	5 << 1 = 10