Day 3 – Python Operators & Expressions

Course: MAHAKTECH Python Programming (Basic to Advanced)

Topics Covered Today:

- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- Logical Operators
- Identity Operators
- Membership Operators
- Operator Precedence (PEMDAS)
- Expressions vs Statements

1️⃣ Arithmetic Operators

Definition:

Arithmetic operators are symbols that perform **mathematical operations** like addition, subtraction, multiplication, and division.

Operator	Meaning	Example	Output
`+`	Addition	`3 + 5`	`8`
`-`	Subtraction	`10 - 4`	`6`
`*`	Multiplication	`2 * 3`	`6`
`/`	Division (float)	`10 / 3`	`3.3333`
`//`	Floor Division	`10 // 3`	`3`
`%`	Modulus (remainder)	`10 % 3`	`1`
`**`	Exponent	`2 ** 3`	`8`

Example Code:

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

2️⃣ Assignment Operators

Definition:

Assignment operators **store or update values in variables**.

Operator	Meaning	Example
`=`	Assign value	`x = 5`
`+=`	Add and assign	`x += 3 → x = x + 3`
`-=`	Subtract and assign	`x -= 2 → x = x - 2`
`*=`	Multiply and assign	`x *= 2`
`/=`	Divide and assign	`x /= 5`
`//=`	Floor divide and assign	`x //= 3`
`%=`	Modulus and assign	`x %= 3`
`**=`	Exponent and assign	`x **= 2`

Example Code:

```
x = 10
x += 5
x *= 2
print(x)  # 30
```

3️⃣ Comparison Operators

Definition:

Comparison operators **compare two values** and return **True** or **False**.

Operator	Meaning	Example	Output
`==`	Equal to	`5 == 5`	True
`!=`	Not equal	`5 != 3`	True
`>`	Greater than	`10 > 5`	True
`<`	Less than	`3 < 7`	True
`>=`	Greater or equal	`5 >= 5`	True
`<=`	Less or equal	`4 <= 7`	True

Example Code:

```
a = 10
b = 5
print(a > b)  # True
print(a == b) # False
```

4️⃣ Logical Operators

Definition:

Logical operators **combine multiple conditions** to produce a Boolean result.

Operator	Meaning	Example	Output
`and`	True if both True	`True and False`	False
`or`	True if at least one True	`True or False`	True
`not`	Reverse value	`not True`	False

Example Code:

```
x = True
y = False
print(x and y)  # False
```

```
print(x or y)   # True
print(not x)    # False
```

5️⃣ Identity Operators

Definition:

Identity operators **check whether two variables point to the same object** in memory.

Operator	Meaning	Example	Output
`is`	True if same object	`a is b`	True/False
`is not`	True if not same object	`a is not b`	True/False

Example Code:

```
a = [1,2,3]
b = [1,2,3]
print(a == b)   # True (values same)
print(a is b)   # False (different objects)
```

6️⃣ Membership Operators

Definition:

Membership operators **check if a value exists in a sequence** (string, list, tuple, set, dictionary).

Operator	Meaning	Example	Output
`in`	True if element exists	`'a' in 'apple'`	True
`not in`	True if element does not exist	`'z' not in 'apple'`	True

Example Code:

```
text = "mahaktech"
print("m" in text)    # True
print("x" not in text) # True
```

7️⃣ Operator Precedence (PEMDAS)

Definition:

Operator precedence defines **the order in which Python evaluates operations** in an expression.

PEMDAS Rule:

Parentheses → Exponent → Multiplication/Division → Addition/Subtraction

Example Code:

```
print(2 + 3 * 4)      # 14
print((2 + 3) * 4)    # 20
print(2 ** 3 ** 2)    # 512
print(100 / 5 * 2)    # 40.0
```

Tips for Students:

- Use parentheses to **control the order**.
- Multiplication/division happens **before addition/subtraction**.
- If operators have same precedence, Python evaluates **left to right**.

8️⃣ Expressions vs Statements

Definition:

- **Expression:** Code that **produces a value**.
- **Statement:** Code that **performs an action**.

Concept	Example
Expression	`3 + 5` , `x * 2` , `10 > 5`
Statement	`x = 5` , `print(x)`

Example Code:

```
# Expression
print(5 + 3)     # 8

# Statement
x = 5 + 3
print("x =", x)  # x = 8
```

Tip:

> Expression → “What is this?”
>
> Statement → “Do this!”