Assignment Operator

A Comprehensive Tutorial

Table of contents

01

Basic Assignment

Basic Assignment

The **assignment operator** `=` in Python is used to **assign the value** on the right side of the operator to the variable on the left side.

Syntax:

```
variable_name = value
```

Example:

```
x = 10
name = “Alice”
```

In the above examples, **x** is assigned the value **10**, and **name** is assigned the string **"Alice"**

02

Chained Assignment

Chained Assignment

Python enables you to **assign a single value to multiple variables simultaneously** using chained assignment.

Syntax:

```
a = b = c = value
```

Example:

```
x = y = z = 5
```

In this example, `x`, `y`, and `z` are all assigned the value `5`.

03

Multiple Assignment

Multiple Assignment

Python allows **multiple variables** to be **assigned** in a **single statement**. This is often used for **swapping values** or **assigning multiple values** at once.

Syntax:

```
var1, var2, var3 = val1, val2, val3
```

Example:

```
a, b, c = 1, 2, 3
```

In this example, `**a**` is assigned `**1**`, `**b**` is assigned `**2**`, and `**c**` is assigned `**3**`.

Use Case of Multiple Assignment

Swapping Values

Multiple assignment is particularly useful for **swapping values without needing a temporary variable**.

Syntax:

```
a, b = b, a
```

04

Unpacking

Unpacking

Python supports **unpacking** of iterable objects (like lists or tuples) into **multiple variables**.

Syntax:

```
var1, var2, var3 = iterable
```

Example:

```
numbers = (1, 2, 3)
a, b, c = numbers
```

Here, `a` will be `1`, `b` will be `2`, and `c` will be `3`.

Extended Unpacking

Python also allows for extended unpacking, where **one variable can capture multiple values.**

Syntax:

```
var1, *var2, var3 = iterable
```

Example:

```
a, *b, c = [1, 2, 3, 4, 5]
```

Here, `a` is `1`, `c` is `5`, and `b` captures the middle values `[2, 3, 4]`.

05

Augmented Assignment

Augmented Assignment

Python provides a shorthand for operations followed by assignment, known as augmented assignment. These operators **combine an operation with an assignment.**

Syntax:

```
variable op= value
```

Example:

```
x = 10
x += 5  #Equivalent to x = x + 5
```

After this, `**x**` will be `**15**`.

Common Operators

		X = 10
+=	Add and assign	`X += 5 → 15`
-=	Subtract and assign	`X -= 5 → 5`
***=**	Multiply and assign	`X *= 5 → 50`
/=	Divide and assign	`X /= 5 → 2.0`
//=	Floor Divide and assign	`X //= 5 → 2`
%=	Modulus and assign	`X %= 5 → 0`
****=**	Exponentiate and assign	`X **= 5 → 100000`

Common Operators

		X = 10
&=	Bitwise AND and assign	X &= 5 → 0
|=	Bitwise OR and assign	X |= 5 → 15
^=	Bitwise XOR and assign	X ^= 5 → 15
<<=	Bitwise Left Shift and assign	X <<= 1 → 20
>>=	Bitwise Right Shift and assign	X >>= 1 → 5

Knowledge Reinforcement

Basic Assignment

Objective:

- Assign the value **25** to a variable named **age**.
- Assign the string "**Python Programming**" to a variable named **course**.
- **Print** both age and course.

Code:

Basic Assignment

Objective:

- Assign the value **25** to a variable named **age**.
- Assign the string "**Python Programming**" to a variable named **course**.
- **Print** both age and course.

Code:

```
age = 25
course = “Python Programming”
print(age, course)
```

Chained Assignment

Objective:

- **Assign** the value **50** to three variables **a, b**, **and c** using chained assignment.
- **Print** the values of a, b, and c.

Code:

Chained Assignment

Objective:

- **Assign** the value **50** to three variables **a, b, and c** using chained assignment.
- **Print** the values of a, b, and c.

Code:

```
a = b = c = 50
print(a, b, c)
```

Multiple Assignment

Objective:

- Assign the values **5, 10, and 15** to variables **x, y, and z** in a single line.
- Print the **sum** of x, y, and z.

Code:

Multiple Assignment

Objective:

- Assign the values **5, 10, and 15** to variables **x, y, and z** in a single line.
- Print the **sum** of x, y, and z.

Code:

```
x, y, z = 5, 10, 15
print(“Sum: “, x+y+z)
```

Swapping Values

Objective:

- Assign **100** to variable **m** and **200** to variable **n**.
- **Swap** the values of **m and n** using a single line of code.
- **Print** the values of **m and n** after swapping.

Code:

Swapping Values

Objective:

- Assign **100** to variable **m** and **200** to variable **n**.
- **Swap** the values of **m and n** using a single line of code.
- **Print** the values of **m and n** after swapping.

Code:

```
m = 100
n = 200
m, n = n, m
print(m, n)
```

Augmented Assignment

Objective:

- Start with a variable **total** assigned the value **0**.
- **Add 50** to **total** using an augmented assignment operator.
- **Subtract 20 from total** using an augmented assignment operator.
- **Multiply total by 3** using an augmented assignment operator.
- **Divide total by 2** using an augmented assignment operator.
- Print the **final value** of total

Code:

Augmented Assignment

Objective:

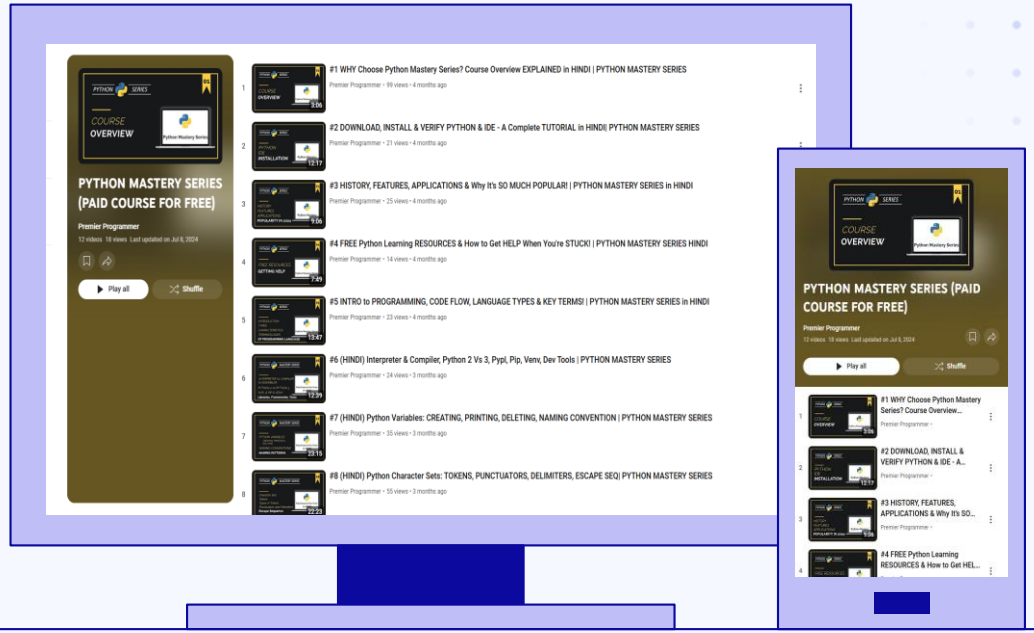
- Start with a variable **total** assigned the value **0**.
- **Add 50** to **total** using an augmented assignment operator.
- **Subtract 20 from total** using an augmented assignment operator.
- **Multiply total by 3** using an augmented assignment operator.
- **Divide total by 2** using an augmented assignment operator.
- Print the **final value** of total

Code:

```
Total = 0
Total += 50
Total -= 20
Total *= 3
Total /=2
print(Total)
```

WATCH

Level up your coding with each episode in this focused Python series.



Next Video!

Comparison Operator – In Depth