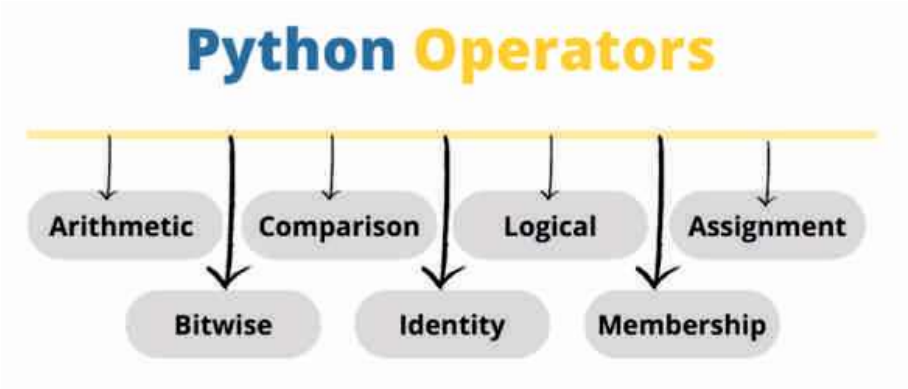
Chapter-4
PYTHON DATA OPERATORS

In the last chapter we learnt about

- **Variables and Values**

We also used **"="** **operator** to **assign** a **value** into a **variable**. Operators are used by every programming language.

In this chapter we will look into some more common operators used in Python.



Definitions –

- **Operators** are used to ***perform operations*** on **variables** and **values**.
- These are ***standard symbols*** used for the **purpose of operations**.
- Arithmetic, Comparison, Logical, and Assignment **operators** are the most commonly used. However, there are other types as well.

Arithmetic Operators

- Arithmetic operators are used to performing mathematical operations like ***addition, subtraction, multiplication, and division***.
- In other words, **Arithmetic operators** are used with **numeric values** to perform common **mathematical operation**
- In other words, with ***int or float*** data types

Operator	Description	Syntax
+	Addition: adds two operands	x + y
–	Subtraction: subtracts two operands	x – y
*	Multiplication: multiplies two operands	x * y
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	x // y
%	Modulus: returns the remainder when the first operand is divided by the second	x % y
**	Power: Returns first raised to power second	x ** y

Arithmetic Operators Examples with variables a=9 and b = 4

```
# Addition of numbers
add = a + b

# Subtraction of numbers
sub = a - b

# Multiplication of number
mul = a * b

# Division(float) of number
div1 = a / b

# Division(floor) of number
div2 = a // b

# Modulo of both number
mod = a % b

# Power
p = a ** b
```

```
# Lets run this code

# print results
print(add)
print(sub)
print(mul)
print(div1)
print(div2)
print(mod)
print(p)
```

Output

```
13
5
36
2.25
2
1
6561
```

Comparison Operators

- **Comparison or Relational** operators are used to **compare** the values.
- It either returns **True or False** according to condition
- Comes handy in **control structures** like ***condition, iteration and sequence.***

Operator	Description	Syntax
>	Greater than: True if the left operand is greater than the right	x > y
<	Less than: True if the left operand is less than the right	x < y
==	Equal to: True if both operands are equal	x == y
!=	Not equal to – True if operands are not equal	x != y
>=	Greater than or equal to True if the left operand is greater than or equal to the right	x >= y
<=	Less than or equal to True if the left operand is less than or equal to the right	x <= y

Comparison Operators Examples with variables a=13 and b = 33

```
# a > b is False
print(a > b)

# a < b is True
print(a < b)

# a == b is False
print(a == b)

# a != b is True
print(a != b)

# a >= b is False
print(a >= b)

# a <= b is True
print(a <= b)
```

Output

```
False
True
False
True
False
True
```

Assignment Operators

- Assignment operators are used to **assign** values to the variables.

Operator	Description	Syntax
=	Assign value of right side of expression to left side operand	x = y + z
+=	Add AND: Add right-side operand with left side operand and then assign to left operand	a+=b a=a+b
-=	Subtract AND: Subtract right operand from left operand and then assign to left operand	a-=b a=a-b
*=	Multiply AND: Multiply right operand with left operand and then assign to left operand	a*=b a=a*b
/=	Divide AND: Divide left operand with right operand and then assign to left operand	a/=b a=a/b
%=	Modulus AND: Takes modulus using left and right operands and assign the result to left operand	a%=b a=a%b
//=	Divide(floor) AND: Divide left operand with right operand and then assign the value(floor) to left operand	a//=b a=a//b
**=	Exponent AND: Calculate exponent(raise power) value using operands and assign value to left operand	a**=b a=a**b

```
# Assign value
b = a
print(b)

# Add and assign value
b += a
print(b)

# Subtract and assign value
b -= a
print(b)

# multiply and assign
b *= a
print(b)
```

Assignment Operators Examples with

a = 10

Output

```
10
20
10
100
```

Logical Operators

- Logical operators are used to check the **Logical AND**, **Logical OR**, and **Logical NOT** operations
- It is used to combine conditional statements.

Operator	Description	Syntax
and	Logical AND: True if both the operands are true	x and y
or	Logical OR: True if either of the operands is true	x or y
not	Logical NOT: True if the operand is false	not x

Logical Operators Examples

```
# Examples of Logical Operator
a = True
b = False

# Print a and b is False
print(a and b)

# Print a or b is True
print(a or b)

# Print not a is False
print(not a)
```

Output

```
False
True
False
```

Data Operator Examples

Example 1: Simple Arithmetic Calculation

```makefile
num1 = 10
num2 = 5
result = num1 + num2
print("The sum of", num1, "and", num2, "is", result)
```

Output

```python
The sum of 10 and 5 is 15
```

Explanation:

- In this example, we **declare two variables** `num1` and `num2` and **assign** them the **values** 10 and 5 respectively.
- We then perform the addition operation using the `+` operator and **store** the result in the variable `result`.
- Finally, we use the `print()` function to display a message with the values and the sum.

Example 2: Using Logical Operators

```python
x = True
y = False
print("x AND y:", x and y)
print("x OR y:", x or y)
print("NOT x:", not x)
```

Output

```yaml
x AND y: False
x OR y: True
NOT x: False
```

Explanation: In this example, we **define two variables** `x` and `y` with boolean values. We use the `and`, `or`, and `not` **logical operators** to perform logical operations on these variables. The `print()` function is used to display the results of these operations.

Example 3: Using Assignment Operators

```makefile
num = 10
num += 5
print("Updated num:", num)
```

Output

```yaml
Updated num: 15
```

Explanation: In this example, we declare a variable `num` and assign it the value `10`. We then use the `+=` assignment operator to add `5` to the existing value of `num`. Finally, we **print** the updated value of `num`.

Example 4: Comparison Operators

```
num1 = 10
num2 = 5
print("num1 > num2:", num1 > num2)
print("num1 < num2:", num1 < num2)
print("num1 == num2:", num1 == num2)
```

Output

```
num1 > num2: True
num1 < num2: False
num1 == num2: False
```

Explanation: In this example, we define two variables `num1` and `num2` with different values. We use the comparison operators `>`, `<`, and `==` to compare the values of these variables. The results of these comparisons are then printed using the `print()` function.

Example 5: Working with Strings

```
name = input("Enter your name: ")
greeting = "Hello, " + name + "!"
print(greeting)
```

Output

```
Enter your name: Alice
Hello, Alice!
```

Explanation: In this example, we use the `input()` function to ask the user to enter their name. The value entered by the user is stored in the variable `name`. We then concatenate the string "Hello, ", the value of `name`, and the exclamation mark to create the `greeting` string. Finally, we print the greeting using the `print()` function.

Example 6: Finding the Average of Three Numbers

```python
# Get three numbers from the user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))

# Calculate the average
average = (num1 + num2 + num3) / 3

# Display the average
print("Average:", average)
```

Output

```
Enter the first number: 4
Enter the second number: 6
Enter the third number: 8
Average: 6.0
```

Explanation:

In this program the user is asked to enter three numbers using the `input()` function. Each input is **converted** to a floating-point value using the `float()` function and **stored** in variables `num1`, `num2`, and `num3`.

The **average** is calculated by **adding** the three numbers together and **dividing** the sum by 3. The result is stored in the variable `average`.

Finally, the program prints the average using the `print()` function, along with the label "Average:". The value of `average` is displayed as the output.

Example 7: Simple Interest Calculation

```
principal = float(input("Enter the principal amount: "))
rate = float(input("Enter the interest rate: "))
time = float(input("Enter the time in years: "))

simple_interest = (principal * rate * time) / 100

print("Simple Interest:", simple_interest)
```

Output

```
Enter the principal amount: 1000
Enter the interest rate: 5
Enter the time in years: 2
Simple Interest: 100.0
```

Explanation: This program calculates the simple interest based on the principal amount, interest rate, and time entered by the user.

Example 8: Area of a Circle

```
radius = float(input("Enter the radius of the circle: "))

area = 3.14159 * radius**2

print("Area of the circle:", area)
```

Output

```
Enter the radius of the circle: 5
Area of the circle: 78.53975
```

Explanation: This program calculates the area of a circle based on the radius entered by the user using the formula π * r^2.

Practice Programs

1

Program to calculate the area of a triangle given the base and height. You can either define the base and height or ask the user.

2

Program to convert kilometers to miles. In this case, ask the user to enter the value in km which you will convert and display at the screen

3

Program to calculate the sum of the digits of a three-digit number

4

Program to calculate the average of four test scores. Ask the user to enter the scores and store them in different variables

5

Program to calculate the area of a circle given the radius

6

Program to concatenate two strings using the + operator

7

Program to calculate the remainder when dividing two numbers