

Welcome to python Course for Artificial Intelligence

This course consists of,

1. Python Basic
2. Python Intermediate
3. Python Advance

**Python Basic:**

**Variables**: Containers for storing data values.

* x = 5
* y = "Hello"

**Data Types:** Categories of data that tell the interpreter how the programmer intends to use the data.

int: Integer numbers

**Num =**

**Pi =**

**Name =**

**Status =**

* num = 10

float: Floating point numbers

* pi = 3.14

str: String (text)

* name = "Alice"

bool: Boolean (True or False)

* is\_student = True

list: Ordered collection

* numbers = [1, 2, 3]

tuple: Ordered, immutable collection

* point = (10, 20)

dict: Unordered, key-value pairs

* person = {"name": "Bob", "age": 25}

**Program 1: Simple Calculator**

This program takes two numbers and performs basic arithmetic operations.

**v# Simple Calculator**

**num1 = 10**

**num2 = 5**

**addition = num1 + num2**

**subtraction = num1 - num2**

**multiplication = num1 \* num2**

**division = num1 / num2**

**print(f"Addition: {addition}")**

**print(f"Subtraction: {subtraction}")**

**print(f"Multiplication: {multiplication}")**

**print(f"Division: {division}")**



**Program 2: Greeting Message**

This program takes a user's name and prints a personalized greeting.

**# Greeting Message**

**name = "Alice"**

**greeting = f"Hello, {name}! Welcome to Python programming."**

**print(greeting)**

**Program 3: List Operations**

This program demonstrates basic list operations like appending, removing, and accessing elements.

**# List Operations**

**fruits = ["apple", "banana", "cherry"]**

**# Adding an element**

**fruits.append("orange")**

**# Removing an element**

**fruits.remove("banana")**

**# Accessing elements**

**first\_fruit = fruits[0]**

**last\_fruit = fruits[-1]**

**print(f"Fruits List: {fruits}")**

**print(f"First Fruit: {first\_fruit}")**

**print(f"Last Fruit: {last\_fruit}")**



**Variables Rules:**

1 Must start with a letter or underscore (\_): Variable names cannot start with a number.

* Valid: variable, \_variable
* Invalid: 1variable

2 Can only contain alphanumeric characters and underscores: No spaces or special characters (except underscore).

* Valid: variable\_1, var\_name
* Invalid: variable-name, var name

3 Case-sensitive: Variable and variable are different variables.

* Example: Variable ≠ variable

4 Cannot be a reserved keyword: You cannot use Python keywords as variable names (e.g., if, else, while, etc.).

* Invalid: if, for, True

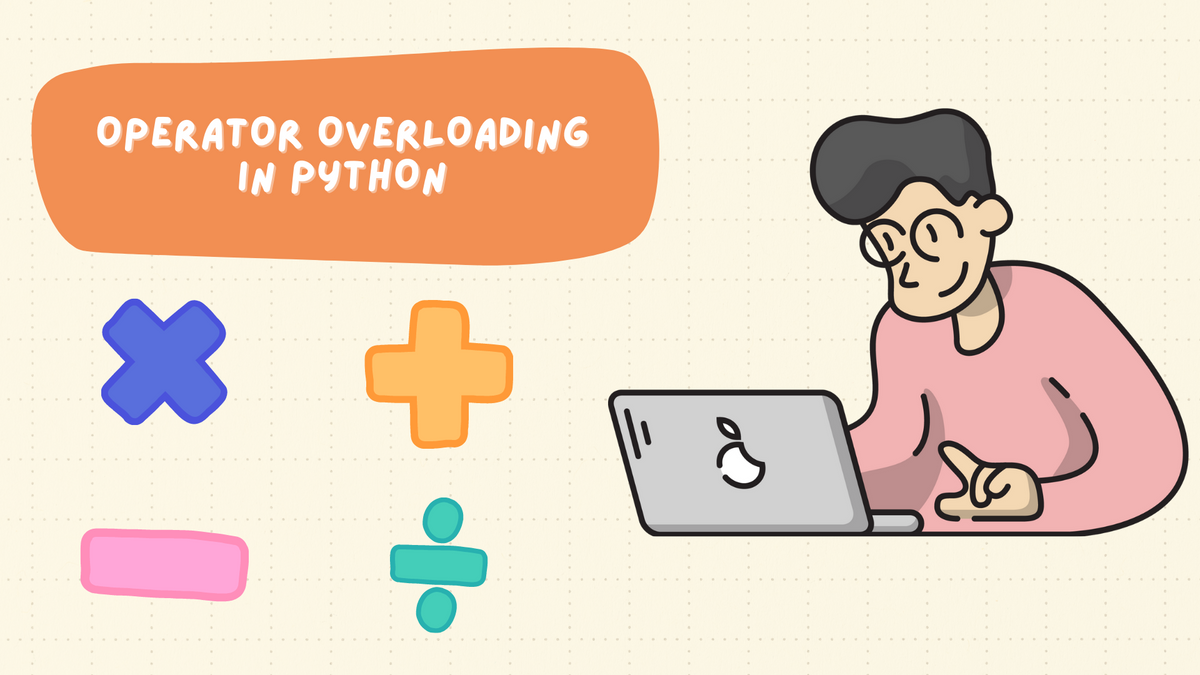
5 Should be descriptive: While not a strict rule, it's good practice to use meaningful names that describe the purpose of the variable.

* Example: count, total\_sum

**Python Basic**

**Operators In python**

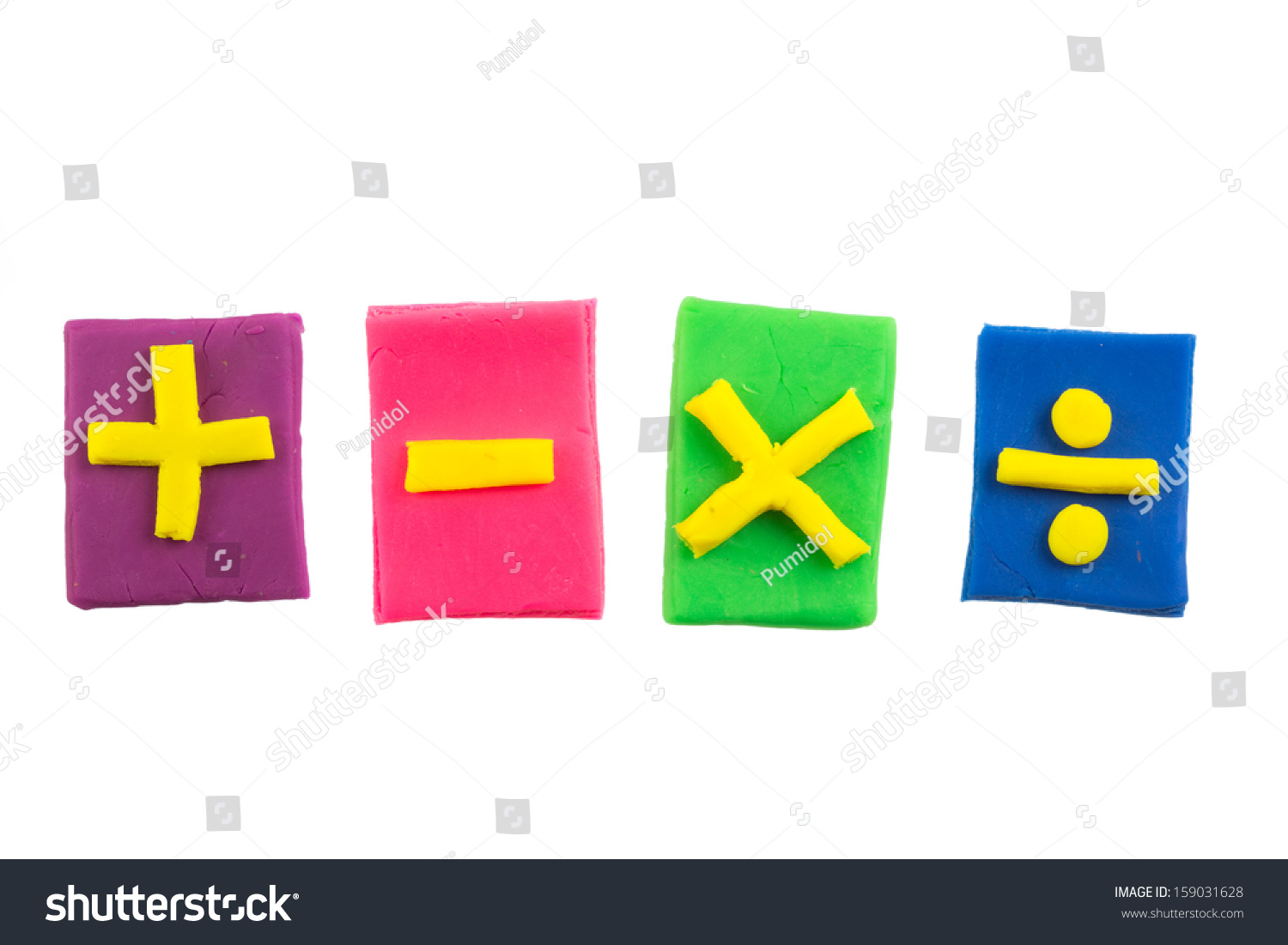
In Python, an operator is a symbol that performs an operation on one or more operands, like + for addition.



Example: 3 + 5 uses the + operator to add the numbers 3 and 5.

Usage: Operators simplify code by performing mathematical, logical, or bitwise operations concisely.

**1 Arithmetic Operators:**





Program:

**# Define variables**

**a = 15**

**b = 4**

**# Perform arithmetic operations**

**sum\_result = a + b**

**diff\_result = a - b**

**product\_result = a \* b**

**quotient\_result = a / b**

**floor\_div\_result = a // b**

**modulus\_result = a % b**

**exponent\_result = a \*\* b**

**# Display results**

**print(f"The sum of {a} and {b} is {sum\_result}")**

**print(f"The difference between {a} and {b} is {diff\_result}")**

**print(f"The product of {a} and {b} is {product\_result}")**

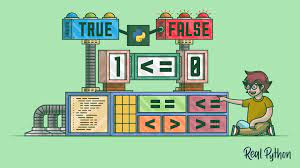
**print(f"The quotient of {a} divided by {b} is {quotient\_result}")**

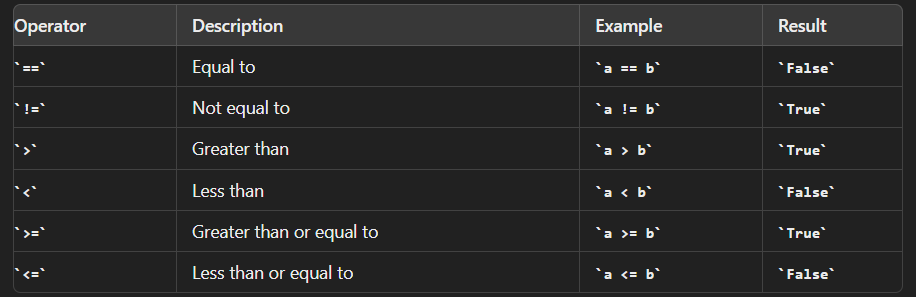
**print(f"The floor division of {a} by {b} is {floor\_div\_result}")**

**print(f"The modulus of {a} and {b} is {modulus\_result}")**

**print(f"The result of {a} raised to the power of {b} is {exponent\_result}")**

**2 Comparison Operators:**





Program

**# Define variables**

**a = 15**

**b = 4**

**# Perform comparison operations**

**equal\_result = (a == b)**

**not\_equal\_result = (a != b)**

**greater\_than\_result = (a > b)**

**less\_than\_result = (a < b)**

**greater\_equal\_result = (a >= b)**

**less\_equal\_result = (a <= b)**

**# Display results**

**print(f"Is {a} equal to {b}? {equal\_result}")**

**print(f"Is {a} not equal to {b}? {not\_equal\_result}")**

**print(f"Is {a} greater than {b}? {greater\_than\_result}")**

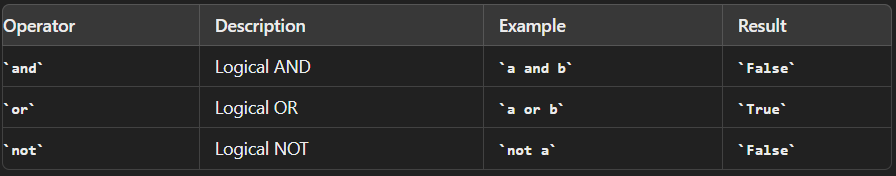
**print(f"Is {a} less than {b}? {less\_than\_result}")**

**print(f"Is {a} greater than or equal to {b}? {greater\_equal\_result}")**

**print(f"Is {a} less than or equal to {b}? {less\_equal\_result}")**

**3 Logical Operators:**





Program

**# Define variables**

**a = True**

**b = False**

**# Perform logical operations**

**and\_result = a and b**

**or\_result = a or b**

**not\_a\_result = not a**

**not\_b\_result = not b**

**# Display results**

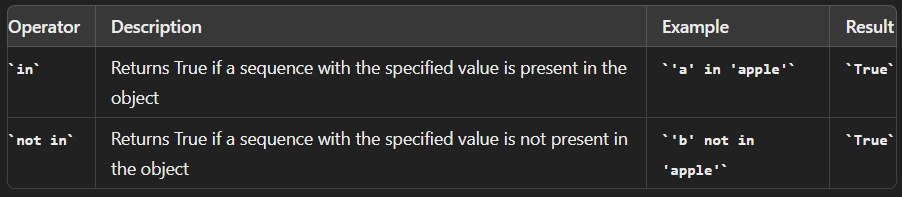
**print(f"The result of {a} and {b} is {and\_result}")**

**print(f"The result of {a} or {b} is {or\_result}")**

**print(f"The result of not {a} is {not\_a\_result}")**

**print(f"The result of not {b} is {not\_b\_result}")**

**4 Membership Operators**



**# Define variables**

**sequence = [1, 2, 3, 4, 5]**

**element1 = 3**

**element2 = 6**

**# Perform membership operations**

**in\_result1 = element1 in sequence**

**in\_result2 = element2 in sequence**

**not\_in\_result1 = element1 not in sequence**

**not\_in\_result2 = element2 not in sequence**

**# Display results**

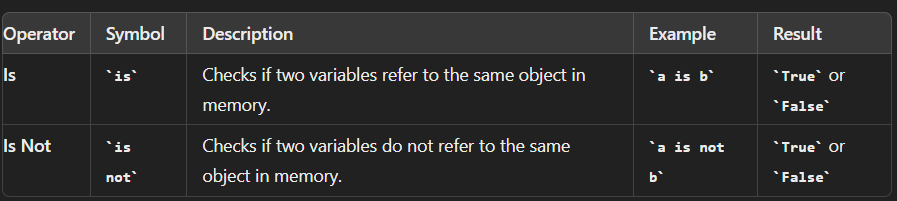
**print(f"Is {element1} in {sequence}? {in\_result1}")**

**print(f"Is {element2} in {sequence}? {in\_result2}")**

**print(f"Is {element1} not in {sequence}? {not\_in\_result1}")**

**print(f"Is {element2} not in {sequence}? {not\_in\_result2}")**

**5 Identity Operators**



**# Example variables**

**a = [1, 2, 3]**

**b = [1, 2, 3]**

**c = a**

**# Using `is`**

**print(a is b) # False, as a and b are different objects**

**print(a is c) # True, as a and c refer to the same object**

**# Using `is not`**

**print(a is not b) # True, as a and b are different objects**

**print(a is not c) # False, as a and c refer to the same object**