Day-8 Content-Material:

1. Arithmetic Operators

- Definition: These operators perform basic mathematical calculations on numbers.
- Operators:
- Addition (+): Combines values.
- Subtraction (-): Finds the difference.
- Multiplication (*): Calculates product.
- Division (/): Divides and returns a float.
- Integer Division (//): Divides and returns an integer (rounded down).
- Modulus (%): Returns the remainder.
- Exponentiation (**): Raises one number to the power of another.

Examples:

Python

```
x = 10

y = 5


print(x + y)  # Output: 15

print(x - y)  # Output: 5

print(x * y)  # Output: 50

print(x / y)  # Output: 2.0

print(x // y)  # Output: 2

print(x % y)  # Output: 0

print(x ** 2)  # Output: 100
```

2. Comparison Operators

- Definition: These operators compare values and return a boolean result (True or False).
- Operators:
  - > (Greater than)
  - < (Less than)
  - >= (Greater than or equal to)
  - <= (Less than or equal to)
  - == (Equal to)
  - != (Not equal to)
- Examples:

```
Python
x = 10
y = 5

print(x > y)  # Output: True
print(x < y)  # Output: False
print(x >= y)  # Output: True
print(x <= y)  # Output: False
print(x == y)  # Output: False
print(x != y)  # Output: True
```

3. Logical Operators

- Definition: These operators combine multiple conditions and return a boolean result.
- Operators:
  - and (Returns True if both conditions are True)
  - or (Returns True if at least one condition is True)
  - not (Reverses the boolean value)
- Examples:

```
Python
x = 10
y = 5

print(x > 5 and y < 10)  # Output: True
print(x < 5 or y > 3)  # Output: True
print(not (x > 5))  # Output: False
```

4. Bitwise Operators

- Definition: These operators work on the binary representation of numbers.
- Operators:
  - & (Bitwise AND)
  - | (Bitwise OR)
  - ^ (Bitwise XOR)
  - ~ (Bitwise NOT)
  - << (Left Shift)
  - >> (Right Shift)

- Example:

```
x = 10  # Binary: 1010
y = 5   # Binary: 0101

print(x & y)  # Output: 0 (Binary: 0000)
print(x | y)  # Output: 15 (Binary: 1111)
print(x ^ y)  # Output: 15 (Binary: 1111)
print(~x)   # Output: -11 (Binary: 1010 -> Two's complement)
print(x << 2) # Output: 40 (Binary: 101000)
print(x >> 1) # Output: 5 (Binary: 0101)
```

5. Assignment Operators

- Definition: These operators assign values to variables.
- Operators:
  - = (Simple assignment)
  - += (Add and assign)
  - -= (Subtract and assign)
  - *= (Multiply and assign)
  - /= (Divide and assign)
  - //= (Floor divide and assign)
  - %= (Modulo and assign)
  - **= (Exponentiate and assign)

Examples:

x = 10

x += 5 # Equivalent to x = x + 5

print(x) # Output: 15

6. Membership Operators

- Definition: These operators check if a value is a member of a sequence (like a list, tuple, or string).
- Operators:
  - in (Returns True if the value is found in the sequence)
  - not in (Returns True if the value is not found in the sequence)
- Examples:

Python

my_list = [1, 2, 3, 4]

print(3 in my_list) # Output: True

print(5 in my_list) # Output: False

7. Identity Operators

- Definition: These operators check if two variables refer to the same object in memory.
- Operators:
    - is (Returns True if both variables refer to the same object)
    - is not (Returns True if both variables refer to different objects)

Examples:

```
x = [1, 2, 3]

y = x

z = [1, 2, 3]


print(x is y)  # Output: True (x and y refer to the same list object)

print(x is z)  # Output: False (x and z refer to different list objects, even if they have the same values)
```