Operators

there are the following operators :

1. Arithmetic
2. assignment
3. comparison
4. logical
5. identity
6. membership
7. bitwise

1. Arithmetic Operators:

- Perform mathematical operations on numbers.
- Examples: `+` (addition), `-` (subtraction), `*` (multiplication), `/` (division), `//` (floor division), `%` (modulo), `**` (exponentiation)

2. Assignment Operators:

- Assign values to variables.
- Example: `=` (simple assignment), `+=`, `-=`, `*=`, `/=` (combined assignment with operation)

```
x = 5
x += 3  # Equivalent to x = x + 3
print(x)  # Output: 8

y = 10
y -= 2  # Equivalent to y = y - 2
print(y)  # Output: 8

z = 4
z *= 3  # Equivalent to z = z * 3
print(z)  # Output: 12

a = 12
a /= 2  # Equivalent to a = a / 2
print(a)  # Output: 6
```

3. Comparison Operators:

- Compare values and return boolean (True/False) results.
- Examples: `==` (equal to), `!=` (not equal to), `<` (less than), `>` (greater than), `<=` (less than or equal to), `>=` (greater than or equal to)

4. Logical Operators:

- Combine conditional statements.
- Examples: `and` (returns True if both conditions are True), `or` (returns True if at least one condition is True), `not` (inverts the boolean value)

5. Identity Operators:

- Check if objects are the same object in memory.
- Examples: `is` (returns True if objects are the same object), `is not` (returns True if objects are different objects)

6. Membership Operators:

- Check if a value is present in a sequence (like lists, tuples, strings).
- Examples: `in` (returns True if value is found in sequence), `not in` (returns True if value is not found)

7. Bitwise Operators:

- Perform operations on the binary representation of numbers (advanced).
- Examples: `&` (bitwise AND), `|` (bitwise OR), `^` (bitwise XOR), `~` (bitwise NOT), `<<` (left shift), `>>` (right shift)