
# Arithmetic, Logical, and Comparison Operators in Python

John Karuitha, PhD

February 27, 2025

## 1 Arithmetic and Logical Operators in Python

Python provides several **arithmetic** and **logical operators** that help in performing basic calculations and making decisions. Understanding these operators is crucial for writing programs that involve mathematical operations or evaluating conditions.

---

### 1.1 Arithmetic Operators

Arithmetic operators are used to perform basic mathematical operations such as addition, subtraction, multiplication, and division. Here's a list of arithmetic operators:

| Operator | Operation | Example | Result |
|---|---|---|---|
| `+` | Addition | `5 + 3` | `8` |
| `-` | Subtraction | `5 - 3` | `2` |

| Operator | Operation | Example | Result |
|---|---|---|---|
| * | Multiplication | 5 * 3 | 15 |
| / | Division | 5 / 3 | 1.6667 |
| // | Floor Division | 5 // 3 | 1 |
| % | Modulus (remainder) | 5 % 3 | 2 |
| ** | Exponentiation | 5 ** 3 | 125 |

### 1.1.1 Examples:

```
# Addition
result = 5 + 3
print(result)  # Output: 8

# Subtraction
result = 5 - 3
print(result)  # Output: 2

# Multiplication
result = 5 * 3
print(result)  # Output: 15

# Division
result = 5 / 3
print(result)  # Output: 1.6666666666666667

# Floor Division
result = 5 // 3
print(result)  # Output: 1

# Modulus (Remainder)
result = 5 % 3
print(result)  # Output: 2

# Exponentiation
result = 5 ** 3
print(result)  # Output: 125
```

---

### 1.1.2 Order of Operations (PEMDAS)

In Python, arithmetic operations are carried out based on a hierarchy known as **order of operations**, often remembered as **PEMDAS**: - **P**arentheses `()` - **E**xponentiation `**` - **M**ultiplication `*` and **D**ivision `/`, `//` - **A**ddition `+` and **S**ubtraction `-`

Python will first evaluate any expressions inside **parentheses**, then handle **exponentiation**, followed by **multiplication** and **division**, and finally, **addition** and **subtraction**.

**Example of Order of Operations:**

```
# Without parentheses
result = 5 + 3 * 2
print(result)  # Output: 11 (Multiplication before addition)

# With parentheses to change the order
result = (5 + 3) * 2
print(result)  # Output: 16 (Parentheses evaluated first)
```

---

## 1.2 Logical Operators

Logical operators are used to combine multiple conditions and return a Boolean value (`True` or `False`). These are often used in decision-making, like `if` statements.

| Operator | Description | Example | Result |
|---|---|---|---|
| `and` | Returns `True` if both conditions are `True` | `(5 > 3) and (2 < 4)` | `True` |
| `or` | Returns `True` if at least one condition is `True` | `(5 > 3) or (2 > 4)` | `True` |
| `not` | Reverses the result, `True` becomes `False` and vice versa | `not (5 > 3)` | `False` |

### 1.2.1 Examples:

```
# and operator (both conditions must be True)
result = (5 > 3) and (2 < 4)
print(result)  # Output: True

# or operator (one condition can be True)
result = (5 > 3) or (2 > 4)
print(result)  # Output: True

# not operator (reverses the condition)
result = not (5 > 3)
print(result)  # Output: False
```

---

## 2 Comparison Operators

These operators are used to compare two values, and they return a Boolean result (`True` or `False`).

| Operator | Meaning | Example | Result |
|---|---|---|---|
| `==` | Equal to | `5 == 5` | `True` |
| `!=` | Not equal to | `5 != 3` | `True` |
| `>` | Greater than | `5 > 3` | `True` |
| `<` | Less than | `5 < 3` | `False` |
| `>=` | Greater than or equal to | `5 >= 5` | `True` |
| `<=` | Less than or equal to | `5 <= 3` | `False` |

### 2.1 Examples:

```
# Equal to
print(5 == 5)  # Output: True

# Not equal to
print(5 != 3)  # Output: True

# Greater than
print(5 > 3)  # Output: True
```

```
# Less than
print(5 < 3)  # Output: False
```

---

## 3 Combining Arithmetic and Logical Operators:

You can combine both arithmetic and logical operators to create complex expressions.

### 3.1 Example:

```
# Check if the sum of 5 and 3 is greater than 7
result = (5 + 3) > 7
print(result)  # Output: True

# Check if 5 is greater than 3 and 10 divided by 2 equals 5
result = (5 > 3) and (10 / 2 == 5)
print(result)  # Output: True
```

---

## 4 Conclusion

Understanding and mastering **arithmetic** and **logical operators** is fundamental for Python programming. Arithmetic operators allow you to perform calculations, while logical operators enable you to make decisions and evaluate multiple conditions. By understanding the **order of operations** (PEMDAS) and combining these operators in complex expressions, you will write more efficient and functional code.

## References