

Day 03 – Arithmetic Operations & Boolean Logic

Learn Python arithmetic operations, variable assignments, and understand boolean expressions and logic. Understanding arithmetic and boolean logic is essential for writing any meaningful Python programs.

Arithmetic Operations: Integers

```
In [1]: print('Addition: ', 1 + 2)
print('Subtraction: ', 2 - 1)
print('Multiplication: ', 2 * 3)
print('Division: ', 4 / 2)
print('Division: ', 6 / 2)
print('Division: ', 7 / 2)
print('Division without the remainder: ', 7 // 2)
print('Modulus: ', 3 % 2)
print('Division without the remainder: ', 7 // 3)
print('Exponential: ', 3 ** 2)

Addition: 3
Subtraction: 1
Multiplication: 6
Division: 2.0
Division: 3.0
Division: 3.5
Division without the remainder: 3
Modulus: 1
Division without the remainder: 2
Exponential: 9
```

Floating & Complex Numbers

```
In [2]: print('Floating Number, PI:', 3.14)
print('Floating Number, gravity:', 9.81)
print('Complex number:', 1 + 1j)
print('Multiplying complex numbers:', (1 + 1j) * (1 - 1j))

Floating Number, PI: 3.14
Floating Number, gravity: 9.81
Complex number: (1+1j)
Multiplying complex numbers: (2+0j)
```

Arithmetic with Variables

```
In [3]: a = 3
b = 2
total = a + b
diff = a - b
product = a * b
division = a / b
remainder = a % b
floor_division = a // b
exponential = a ** b
print('a + b =', total)
print('a - b =', diff)
print('a * b =', product)
print('a / b =', division)
print('a % b =', remainder)
print('a // b =', floor_division)
print('a ** b =', exponential)
```

```
a + b = 5
a - b = 1
a * b = 6
a / b = 1.5
a % b = 1
a // b = 1
a ** b = 9
```

Geometry & Physics Examples

```
In [4]: radius = 10
area_of_circle = 3.14 * radius ** 2
print('Area of a circle:', area_of_circle)

length = 10
width = 20
area_of_rectangle = length * width
print('Area of rectangle:', area_of_rectangle)

mass = 75
gravity = 9.81
weight = mass * gravity
print('Weight of the object:', weight, 'N')
```

```
Area of a circle: 314.0
Area of rectangle: 200
Weight of the object: 735.75 N
```

Boolean Expressions & Comparisons

```
In [5]: print(3 > 2)
print(3 >= 2)
print(3 < 2)
print(2 < 3)
print(2 <= 3)
print(3 == 2)
print(3 != 2)

print(len('mango') == len('avocado'))
print(len('mango') != len('avocado'))
print(len('mango') < len('avocado'))
print(len('milk') != len('meat'))
print(len('milk') == len('meat'))
print(len('tomato') == len('potato'))
print(len('python') > len('dragon'))
```

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

Boolean Logic & Identity

```
In [6]: print('True == True:', True == True)
print('True == False:', True == False)
print('False == False:', False == False)
```

```
print('True and True:', True and True)
print('True or False:', True or False)
print('1 is 1:', 1 is 1)
print('1 is not 2:', 1 is not 2)
print('A in Asabeneh:', 'A' in 'Asabeneh')
print('B in Asabeneh:', 'B' in 'Asabeneh')
print('coding in sentence:', 'coding' in 'coding for all')
print('a in an:', 'a' in 'an')
print('4 is 2 ** 2:', 4 is 2 ** 2)
```

```
True == True: True
True == False: False
False == False: True
True and True: True
True or False: True
1 is 1: True
1 is not 2: True
A in Asabeneh: True
B in Asabeneh: False
coding in sentence: True
a in an: True
4 is 2 ** 2: True
```

```
<>:6: SyntaxWarning: "is" with 'int' literal. Did you mean "=="?
<>:7: SyntaxWarning: "is not" with 'int' literal. Did you mean "!="?
<>:12: SyntaxWarning: "is" with 'int' literal. Did you mean "=="?
C:\Users\Arman\AppData\Local\Temp\ipykernel_28668\1968896264.py:6: SyntaxWarning: "is" with 'int'
literal. Did you mean "=="?
    print('1 is 1:', 1 is 1)
C:\Users\Arman\AppData\Local\Temp\ipykernel_28668\1968896264.py:7: SyntaxWarning: "is not" with
'int' literal. Did you mean "!="?
    print('1 is not 2:', 1 is not 2)
C:\Users\Arman\AppData\Local\Temp\ipykernel_28668\1968896264.py:12: SyntaxWarning: "is" with 'in
t' literal. Did you mean "=="?
    print('4 is 2 ** 2:', 4 is 2 ** 2)
```

```
In [7]: print(3 > 2 and 4 > 3)
print(3 > 2 and 4 < 3)
print(3 < 2 and 4 < 3)
print(3 > 2 or 4 > 3)
print(3 > 2 or 4 < 3)
print(3 < 2 or 4 < 3)
print(not 3 > 2)
print(not True)
print(not False)
print(not not True)
print(not not False)
```

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