M.M.

# Day 3 RPA Master Bootcamp

## What are Operators

In Python, operators are symbols used to perform operations on variables and values. Python supports various types of operators, such as arithmetic, assignment, comparison, logical operators. Below are some examples of each type of operator:

**Arithmetic Operators:**

These operators are used to perform basic arithmetic operations.

```
number1 = 10
number2 = 3

# Addition
result_add = number1 + number2

# Subtraction
result_sub = number1 - number2

# Multiplication
result_mul = number1 * number2

# Division (returns a floating-point result)
result_div = number1 / number2

# Integer Division (returns an integer
result, discarding the decimal part)
result_floor_div = number1 // number2

# remainder (returns the remainder of the
division)
result_mod = number1 % number2
```

2. **Assignment Operators:**

These operators are used to assign values to variables.

```
number = 5

# assign 10 to number1
number1 = 10

# assign 5 to number2
number2 = 5

# assign the sum of number1 and number2 to a
number1 += number2       # sum = number1 +
number2

print(number1)


# Shortened assignments
number += 2  # Equivalent to x = x + 2
number -= 1  # Equivalent to x = x - 1
number *= 3  # Equivalent to x = x * 3
number /= 2  # Equivalent to x = x / 2
```

**Comparison Operators:**

These operators are used to compare values and return Boolean results (True or False).

```
number1 = 10
number2 = 5

# Equality
result_equal = number1 == number2  # Output:
False

# Not Equal
result_not_equal = number1 != number2  #
Output: True

# Greater Than
result_greater_than = number1 > number2  #
Output: True

# Less Than
result_less_than = number1 < number2  #
Output: False

# Greater Than or Equal To
result_greater_equal = number1 >= number2  #
Output: True

# Less Than or Equal To
result_less_equal = number1 <= number2  #
Output: False
```

**Logical Operators:**

These operators are used for logical operations and return Boolean results.

```
is_true = True
is_false = False
```

```
result_and = is_true and is_false    # 
Output: False
result_or = is_true or is_false    # Output: 
True
result_not_is_true = not is_true    # Output: 
False
```

**BuiltIn Function Python**

Python comes with a rich set of built-in functions that provide essential functionality for performing various tasks. These functions are readily available in the Python interpreter and can be used without the need for any additional imports. Here are some common built-in functions in Python:

**Mathematical Functions:**

abs(): Returns the absolute value of a number.

round(): Rounds a number to the nearest integer or a specified number of decimals.

min(): Returns the smallest item in an iterable or among multiple arguments.

max(): Returns the largest item in an iterable or among multiple arguments.

sum(): Returns the sum of all elements in an iterable.

**Type Conversion Functions:**

int(): Converts a number or a string containing a whole number to an integer.

float(): Converts a number or a string containing a floating-point number to a float.

str(): Converts an object to its string representation.

list(): Converts an iterable (e.g., tuple, string) to a list.

tuple(): Converts an iterable (e.g., list, string) to a tuple.

**Input and Output Functions:**

print(): Prints the specified message or object to the standard output (console).

input(): Reads a line of input from the user via the console.

**String Manipulation Functions:**

len(): Returns the length of a string or the number of items in an iterable.

str.upper(): Converts a string to uppercase.

str.lower(): Converts a string to lowercase.

str.strip(): Removes leading and trailing whitespace from a string.

str.split(): Splits a string into a list of substrings based on a delimiter.

**Data Structure Functions:**

len(): Returns the number of elements in a list, tuple, dictionary, etc.

sorted(): Returns a new sorted list from the elements of an iterable.

range(): Generates a sequence of numbers within a given range.

**Logical and Comparison Functions:**

bool(): Converts a value to its corresponding Boolean representation.

all(): Returns True if all elements of an iterable are true (or if the iterable is empty).

any(): Returns True if any element of an iterable is true.

**File Handling Functions:**

open(): Opens a file and returns a file object for reading, writing, or both.

read(): Reads the contents of a file.

write(): Writes data to a file.

**Miscellaneous Functions:**

dir(): Returns a list of names in the current local scope or a specified object.

help(): Displays information about the Python object or a specific topic.

type(): Returns the type of an object.

These are just a few examples of the many built-in functions Python provides. You can find more details and additional functions in the Python documentation:

https://docs.python.org/3/library/functions.html

**Practice**

```
"""
Operators
Arithmatic Operators
Assignment Operations
Comparison Operator
Logics Operators
Arithmatic Operator

BuiltIn Functions

"""
# Arithmatic Operators
num1 = 10
num2 = 5

# sum +
result_sum = num1 + num2

print("sum operator result ", result_sum)

# subtract -

result_sub = num1 - num2

print("subtract operator result ",
result_sub)

# multiply *

result_multiply = num1 * num2

print("Result of multiply operator ",
result_multiply)

# Assignment Operator

num3 = 20
numb4 = 10

num3 += numb4
print(num3)

num3 -= numb4
print(num3)

# Comparison Operator
num5 = 5
num6 = 6

# print(num5 == num6)
print("Ali" == "Ali")
print(num5 < num6)
print(num5 > num6)

# logical Operator and or
is_true = True
is_false = False

result_and = is_true and is_false
# result_and = is_true & is_false
print("result_and ", result_and)

result_or = is_true or is_false
print("result_or ", result_or)
```

M.M.

```python
name = "muneeb"
age = 30

print(type(age))

print(type(str(age)))

print(len(name))

# result_sum_func = num1 + num2
# print(result_sum_func)

result = sum([num1, num2])
print(result)

print(name.capitalize())
print(name.upper())
print(name.lower())
print(name.islower())
```