Article

Introduction to Math Operations in Python

What is an operator?

An operator is a special symbol that carries out mathematical or logical computation on variables or values. These variables or values are known as operands. We will be focusing on math operators in this article. Without further ado, let's meet these marvelous math operators.

Meet the operators

There are seven basic math operators in Python.

Here are the operations:

Addition
Subtraction
Multiplication
Float Division
Integer Division
Modulus
Exponentiation

Operator	Meaning	Quick Example
+	Add the two operands	a+b
-	Subtracts the right operand from the left operand	a-b
*	Multiply the operands together	a*b
/	Divides the left operand by the right operand. The result is the quotient but, with the digits after the decimal point.	a/b
//	Divides the left operand by the right operand. The result is the quotient but, with the digits removed after the decimal point.	a//b

%	Divides the left operand by the right operand and returns the remainder	a% b
**	The left operand is raised to the power of the right operand	a**b

Next, let us see these operators in full action. The following examples will explore these operators with variables and values separately. Remember to assign values to your variables. Google colabortory (colab) is used for executing the examples.

Addition

Example with variables

```
value_1 = 5 # variable value_1 is assigned to the value 5
value_2 = 6 # variable value_2 is assigned to value 6
result = value_1 + value_2 # variable result is assign to the addition of value_1 and
value_2.
#Which its result will be stored in variable result.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

```
11
```

Example with values

```
9 + 13
```

The output:

```
22
```

Subtraction

Example with variables

```
value_1 = 300 # variable value_1 is assigned to the value 300
value_2 = 7 # variable value_2 is assigned to value 300
result = value_1 - value_2 # variable result is assign to the subtraction of value_1
and value_2.
#Which its result will be stored in variable result.
```

```
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

293

Example with values

```
25 - 17
```

The output:

8

Multiplication

Examples with variables

```
value_1 = 2 # variable value_1 is assigned to the value 2
value_2 = 16 # variable value_2 is assigned to the value 16
result = value_1 * value_2 # variable result is assigned to the multiplication of
value_1 and value_2.
#Which its result will be stored in variable result.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

32

Example with values

```
10 * 11
```

The output:

110

Before moving on to the next operation, I have a few questions to ask and answer.

Questions

1. What is the difference between integer and float data types?
2. What is normal division?
3. What is the difference between float division and integer division?
4. What is floor division and is it the same as integer division in python?

Answers

1. Both integer and float belong to the numeric data type. Integers contain positive and negative numbers without the fractional part. Floats are real numbers with a fractional part.

2. Normal division is known as integer division. It uses the '/' operator.

3. Recall, float division will have the result with both the whole part and fractional part. Integer division will **only** have the result with the whole part. Float division is both the quotient and remainder. Integer division is just the quotient.

4. Floor division is integer division but, it rounds down the answer and returns the quotient. Floor division and integer division are the same.

Float Division

Example with variables

```
value_1 = 100 # variable value_1 is assigned to the value 100
value_2 = 5 # variable value_2 is assigned to the value 5
result = value_1 / value_2 # variable result is assign to the float division of value_
1 and value_2.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

20.0

Example with values

```
7.2/3.1
```

The output:

2.3225806451612905

Note: We will learn a function that will round this number to a specified number of sig figs.

Integer Division

Example with variables

```
value_1 = 7 # variable value_1 is assign to the value 7
value_2 = 3 # variable value_2 is assign to the value 3
result = value_1 // value_2 # variable result is assign to the integer divsion of
value_1 and value_2.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

2

Example with values

```
7.2//3.1
```

The output:

2.0

Modulus

Example with variables

```
value_1 = 36 # variable value_1 is assigned to the value 36
value_2 = 7 # variable value_2 is assigned to the value 7
result = value_1 % value_2 # variable result is assigned to the modulus operation of
value_1 and value_2.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

```
1
```

Example with values

```
43 % 9
```

The output:

```
7
```

Exponentiation

Example with variables

```
value_1 = 2 # variable value_1 is assigned to the value 2
value_2 = 3 # variable value_2 is assigned to the value 3
result = value_1 ** value_2 # variable result is assigned to the value_1 raised by
value_2.
print(result) # This will print the result of value_1 and value_2 which is stored in
variable result.
```

The output:

```
8
```

Example with values

```
3**4
```

The output:

```
81
```

Simple Math Operation Calculator

Let's make a simple calculator with the math operators we covered. In this example, we need user input and if-else statements. The input() function allows for user input and evaluates the type of expression entered. The expression could be an integer type or a string type, for example.

In this case, I want the user to enter two numbers on the terminal. I will notify the user of this at the beginning of running the program. I will use the float() function in this calculator as well. The float() function will allow the user to enter a float pointing number(any number with a decimal point). The user will be able to enter whole number numbers or integers(e.g.,6,1,or 8) or floating-point numbers(e.g., 7.2,3.1,8.4).

I will use the str() function to convert the entered value into a string. The else statement will give a sort of an error message to the user.

I chose to give the following error message:
`Invalid Selection of Operator! Want to try again?...`

Of course, you can always choose what to say to the user. The following code for this calculator is below.

Simple Math Operation Calculator Code

```
print("*****Simple Math Operation Calculator*****")
print("Hello User, Please only enter numbers or an operator into this calculator.")
first_number = input("Enter a number: ") # Asking user for first number
math_operator = input("Select an operator (+, -, *, /,//,%,**): ")
```

```
second_number = input("Enter a second number: ") # Asking user to choose a second
number

first_number = float(first_number)
second_number = float(second_number)


if math_operator == "+":
  result =  first_number + second_number
elif math_operator == "-":
  result = first_number - second_number
elif math_operator == "*":
  result= first_number * second_number
elif math_operator == "/":
  result = first_number / second_number
elif math_operator == "//":
  result = first_number // second_number
elif math_operator == "%":
  result = first_number % second_number
elif math_operator == "**":
  result = first_number ** second_number
else:
   result = "Invalid Selection of Operator! Want to try again?..."


print("The answer is: " + str(result))
```

For simplicity's sake, I will choose the addition operator with numbers 5 and 6. The output will look like this:

*****Simple Math Operation Calculator*****
Hello User, Please only enter numbers or an operator into this calculator. Enter a number: 5
Select an operator (+, -, *, /,//,%,**): +
Enter a second number: 6
The answer is: 11.0

But, what will happen if I enter # as an operator?

*****Simple Math Operation Calculator*****
Hello User, Please only enter numbers or an operator into this calculator. Enter a number: 1
Select an operator (+, -, *, /,//,%,**): #
Enter a second number: 3
The answer is: Invalid Selection of Operator! Want to try again?...

After this, we will look at helpful functions that can aid us in our math endeavors.

Advice and helpful tips

You can use different python functions for rounding, flooring, and ceiling numbers. In addition, we can get the absolute value of a number. We need to import the math module of python to use these functions except for the round() function.The round() function and abs() are built-in functions .

Rounding

The round function takes two arguments. These arguments are numbers and digits. The digits are the number of decimals to use when rounding the number. If you have the round() with just the number, the digits argument is set to zero by default.

Example

```
result = 7.2/3.1
print("The result of 7.2/3.1 is :",result)
print("The rounded result of 7.2/3.1 is:", round( result))
# I want the number of sig figs to be 3. I will specified the number in the function t
his way...
print("The rounded result with 3 sig figs is:",round( result,3))
```

The output:

```
The result of 7.2/3.1 is : 2.3225806451612905
The rounded result of 7.2/3.1 is: 2
The rounded result with 3 sig figs is: 2.323
```

Flooring

We have already seen that integer division and floor division is the same. Both operations use the '//' operator. The floor function takes one argument. The argument is the number or the input value. The floor function will return the floor value of the argument. It rounds the number down to the nearest integer.

Example

```
import math
math.floor(9.8)
```

The output:

```
9
```

Ceiling

The ceil function takes one argument. The argument is the number or the input value. The ceil function will return the ceiling value of the argument. It rounds the number up to the nearest integer.

Example

```
math.ceil(9.8)
```

The output:

```
10
```

Absolute Value

The abs() function takes one argument. The argument can be an integer or a floating-point number.

Example

```
abs(10.3)
```

The output:

10.3

How about we look at two different scenarios?

First scenario:

Ms. Wilson assigned her class extra credit before the final exam. She has a total of 15 students in her classes. The extra credit work only had five questions to answer correctly.

Five of her students got full marks, four of her students missed one question, three of her students missed two questions, and the rest of her class missed three questions.

How should we calculate the extra credit for her students?

Let's use float division and multiplication operators to solve the problem. We are going to divide the number of correct answers by the total number of questions. Then, we will multiply the result by 100. We will do this for each case.

The code is simple and can be implemented such as this below:

```
print((5/5)*100, "%") # All questions answered correctly. Full Marks
print((4/5)*100, "%") # Missed one question
print((3/5)*100, "%") # Missed two questions
print((2/5)*100, "%") # Missed three questions
```

```
100.0 %
80.0 %
60.0 %
40.0 %
```

The chart below describes the number of students corresponding to the extra credit in both percentage and points.

# of Students	Extra Credit(%)	Extra Credit(pts)
5	100	5
4	80	4
3	60	3
3	40	2

Second scenario:

Matthew has a final exam in his Intro to Python class. He wants to find out what numerical grade he needs to get a 91% in the class. The final exam grade is worth 20 % of his final grade. His current grade is 89.78% and the teacher rounds up the final grade.

The three things we need to know are already here. We know Matthew's current grade, desired grade, and the weight of the final exam.

1. Find the weight of the remaining coursework. We must subtract the weight of the final exam from the weight of the remaining coursework.
2. Matthew's current grade should be multiplied by the weight of the remaining coursework in the class. This will result in the weighted value of his current course grade.
3. Matthew's desired grade will be subtracted by the weighted value of his current grade.
4. The result from step three will be divided by the weight of the final exam. This will give us the answer to what numerical grade he needs on the final exam.

Final Grade Calculator Code

```
print("*****Final Grade Calculator*****")
Desired_grade = input("Enter your desired grade: ") # Asking user for the
desired grade
Current_grade = input("Enter your current grade: ") # Asking user for current grade
Weight_finalexam = input("Enter the weight of your final exam: ") # Asking the
user to enter final exam's worth

Desired_grade = float(Desired_grade)
Current_grade = float(Current_grade)
Weight_finalexam = float(Weight_finalexam)

TotalCourse_weight = 100

Weight_remaining = TotalCourse_weight - Weight_finalexam
Weight_currentgrade = Current_grade * (Weight_remaining)/100
Difference = Desired_grade - Weight_currentgrade
```

```
Numerical_grade = Difference / Weight_finalexam
Numerical_grade_percent = round(Numerical_grade,2)* 100
print("You need to get at least a", Numerical_grade_percent,"% on the final to get
your desired grade.")
```

The output should look like this:

*****Final Grade Calculator*****
Enter your desired grade: 91
Enter your current grade: 89.78
Enter the weight of your final exam: 20

You need to get at least a 96.0 % on the final to get your desired grade.

Float division was used because we want the numbers to be float-pointing numbers. This will help us know what the accurate numerical grade should be. The weight_remaining had to be divided by 100 to get its decimal form. In the end, we use the round function to round the number by two sig figs. Then, we multiply by 100 to get the numerical grade in percentage form.

Here, we used several math operators and the round function to solve Matthew's problem.

Summary

We covered the seven basic math operators in python. Alongside, we saw how different functions help our math to have significance and simplicity. You can always go back over the examples and scenarios in this article. Good luck on your python journey.