Operators and Operands

by Sophia

WHAT'S COVERED

In this lesson, you will learn how to use operators to perform mathematical operations on integers and float variables. Specifically, this lesson covers:

1. Common Mathematical Operators

We've seen the use of variables so far but in order to make them useful, we'll want to be able to perform mathematical operations with our integers and floats. Luckily, how we do so in Python is the same regardless of whether it's an integer or a float. You can perform the following basic mathematical operations:

Name	Operator
Add	+
Subtract	-
Multiply	*
Divide	/
Exponent	**

These operators are the same in most programming languages. To use them, we'll use an expression. An **expression** is a combination of values, variables, and operators. A value all by itself is considered an expression, and so is a variable, so the following are all legal expressions (assuming that the variable x has been assigned a value):

EXAMPLE

```
17
x
```

```
x + 17
```

In a script, an expression all by itself doesn't do anything. This is a common source of confusion for beginners.

TRY IT

Directions: For an example, enter the following code into the IDE and run it.

```
x = 10
x + 17
```

As expected, nothing happens.

Let's put this in a print statement instead.

Directions: Now enter this code into the IDE and run it.

```
x = 10
print(x + 17)
```

You should be seeing 27 as the output.

```
27
```

Rather than just output the expression, we would generally set it to a variable and then output the variable.

```
x = 10
result = x + 17
print(result)
```

So we would see an output of 27.

```
27
```

There are some instances where we need to perform operations on the same variable. For example, we may need to add a value to it or deduct a value. The code may look like this.

```
myValue = 2
myValue = myValue + 2
print(myValue)
```

Other Assignment Operators

In Python, there is another way to perform operations on variables using some basic assignment operators. We have already been using the most basic assignment operator, the `=` operator (equal sign). However, there are more available. Here are a few that work with mathematical operations:

Note: The term operand is used to describe any object that is capable of being manipulated.

Operator	Description	Syntax	Example
=	Assigns the value of the right operand to the left operand.	x = 5	x = 5
+=	Add and Assign: Adds the right operand value with the left operand value and then assigns to the left operand.	x += y	5 += 5 will be 10
-=	Subtract AND: Subtracts the right operand value from the left operand value and then assigns to the left operand.	x -= y	5 -= 3 will be 2
*=	Multiply AND: Multiplies the right operand value with the left operand value and then assigns to the left operand.	x *= y	5 *= 5 will be 25
/=	Divide AND: Divides the left operand value by the right operand value and then assigns to the left operand.	x /= y	5 /= 3 will be 1.66666

We can simplify the previous operation of `myValue = myValue + 2` with the add and assign operator like below:

```
myValue = 2
myValue += 2
print(myValue)
```

This works for all of the operators to modify the variable.

TRY IT

Directions: Enter the following code into the IDE and run it.

```
myValue = 2
print(myValue)
myValue += 2
print(myValue)
myValue *= 2
print(myValue)
myValue -= 3
print(myValue)
myValue /= 2
print(myValue)
```

You should see the following output:

```
2
4
8
5
2.5
```

TRY IT

Directions: Try changing the initial `myValue` value. Does each assignment operation make sense? Try changing some of the numeric values in each operation and run the code snippet.

---

2. Exponent and Modulus Operator

The exponent operator looks a bit different, as it uses **. The base comes first, and the exponent after the **. For example, 10^2 is defined and calculated using the following:

10 ** 2

Or, in code using the variables and then output, as shown below:

TRY IT

Directions: Enter the following code into the IDE and run it.

```
base=10
exponent=2
result = base ** exponent
print(result)
```

You should see the following output:

```
100
```

The modulus operator works on integers and yields the remainder when the first operand is divided by the second. In Python, the modulus operator is a percent sign (%). The syntax is the same as for other operators.

TRY IT

Directions: Enter the following code into the IDE and run it.

```
remainder = 7 % 3
print(remainder)
```

You should see the following output:

```
1
```

So, 7 divided by 3 is 2 with 1 leftover.

The modulus operator turns out to be surprisingly useful. For example, you can check whether one number is divisible by another: if x % y is zero, then x is divisible by y.

TRY IT

Directions: Enter the following code into the IDE and run it.

```
remainder = 99 % 3
print(remainder)
```

You should see the following output:

```
0
```

You can also extract the right-most digit or digits from a number. For example, x % 10 yields the right-most digit of x (in base 10). Similarly, x % 100 yields the last two digits.

TRY IT

Directions: Enter the following code into the IDE and run it.

```
remainder = 6849 % 10
print(remainder)

remainder = 6849 % 100
print(remainder)
```

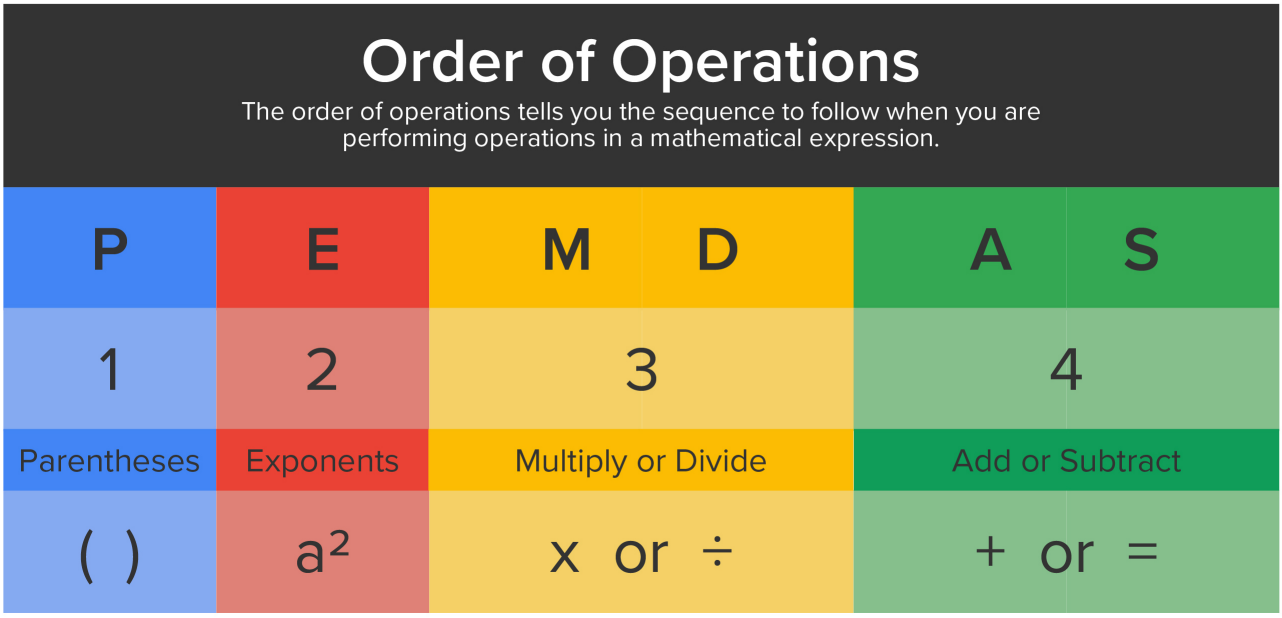
You should see the following output:

```
9
49
```

3. Order of Operations

We can also add multiple different operators in a single statement. When more than one operator appears in an expression, the order of evaluation depends on the rules of precedence.

For mathematical operators, Python follows mathematical convention. The acronym PEMDAS is a useful way to remember the rules.



PEMDAS Definitions	
P (Parentheses)	Parentheses have the highest precedence and can be used to force an expression to evaluate in the order you want. Since expressions in parentheses are evaluated first: 2 * (3-1) is 4, and (1+1)**(5-2) is 8. You can also use parentheses to make an expression easier to read, as in (minute * 100) / 60, even if it doesn't change the result.
E (Exponent)	Exponentiation has the next highest precedence, so: 2**1+1 is 3, not 4, and 3*1**3 is 3, not 27.
MD (Multiplication and Division)	Multiplication and Division have the same precedence, which is higher than Addition and Subtraction, which also have the same precedence, so: 2*3-1 is 5, not 4, and 6+4/2 is 8, not 5.
AS (Addition and Subtraction)	Operators with the same precedence are evaluated from left to right, so: 5-3-1 is 1, not 3, because 5-3 happens first and then 1 is subtracted from 2.

If you're ever unsure about the order of operations, always put parentheses in your expressions to make sure the computations are performed in the order you intend.

TRY IT

Directions: Below are a few examples to work out in the IDE. This will help you see how the order of operations is important. Read the example and enter the example's code, replacing the underlined area with the code that will produce the correct answer. The solutions are at the bottom below the Summary.

Example 1:

Let's say we have 20 pizza slices. You've eaten two and then gave half of the pizza slices to your work. How many pizza slices are left? How would you structure that calculation?

```
pizzaSlices = 20
remainingSlices = your code here
print(remainingSlices)
```

Example 2:

Let's say that you've put in a $300 investment into a new company. The company had to use a third of the initial funds for marketing. Then, they had to take $50 for product development. After that, they were able to successfully deploy their project and gave 5 times the remaining funds to each investor. How much money did they give back to you?

```
investment = 300
money = your code here
print(money)
```

Example 3: How much profit did you make from that initial investment?

```
investment = 300
profit = your code here
print(profit)
```

SUMMARY

In this lesson, we learned the **common mathematical operators** that Python can utilize to perform mathematical operations on integers and float variables. We also discussed how the **exponent and modulus operators** work. We saw that the **order of operations** is very important and how using the acronym PEMDAS is a useful way to remember the rules. Finally, we had an opportunity to apply some calculations that follow the conventional mathematical order of operations.

Best of luck in your learning!

Example Solutions:

Example 1

```
pizzaSlices = 20
remainingSlices = (pizzaSlices - 2) / 2
print(remainingSlices)
```

Example 2

```
investment = 300
money = ((investment * 2/3) - 50) * 5
print(money)
```

Example 3

```
investment = 300
profit = ((investment * 2/3) - 50) * 5 - investment
print(profit)
```

Source: THIS CONTENT AND SUPPLEMENTAL MATERIAL HAS BEEN ADAPTED FROM “PYTHON FOR EVERYBODY” BY DR. CHARLES R. SEVERANCE ACCESS FOR FREE AT www.py4e.com/html3/ LICENSE: CREATIVE COMMONS ATTRIBUTION 3.0 UNPORTED.