Rex
Jones II

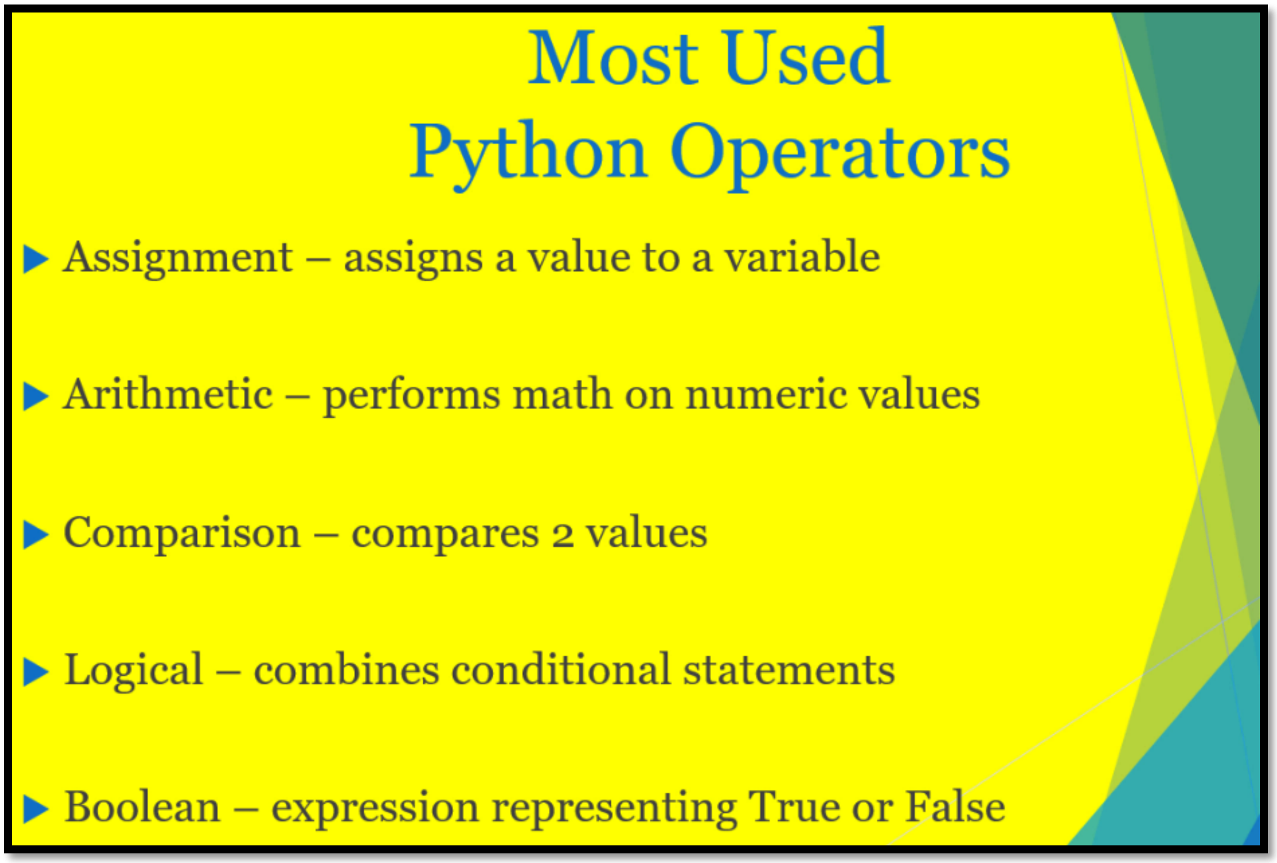
Assignment & Arithmetic Operators

Python Video = https://youtu.be/Y7_jr-QI78A

Operators

In this tutorial session, we are going to look at operators. There are 6 types of Python operators. We have Assignment, Arithmetic, Comparison, Logical, Boolean, and Bitwise. The Identity and Membership operators are a subset of a Boolean operator. We are going to focus on the most used operators:

Assignment, Arithmetic, Comparison, Logical, and Boolean. The purpose of an Assignment Operator is to assign a value to the variable name. Arithmetic Operators perform a math operation on numeric values. A Comparison Operator is used to compare 2 values. and the Logical Operators combine statements that have a condition. Last but not least, are the Boolean operators which represent an expression that returns True or False.



Most Used Python Operators

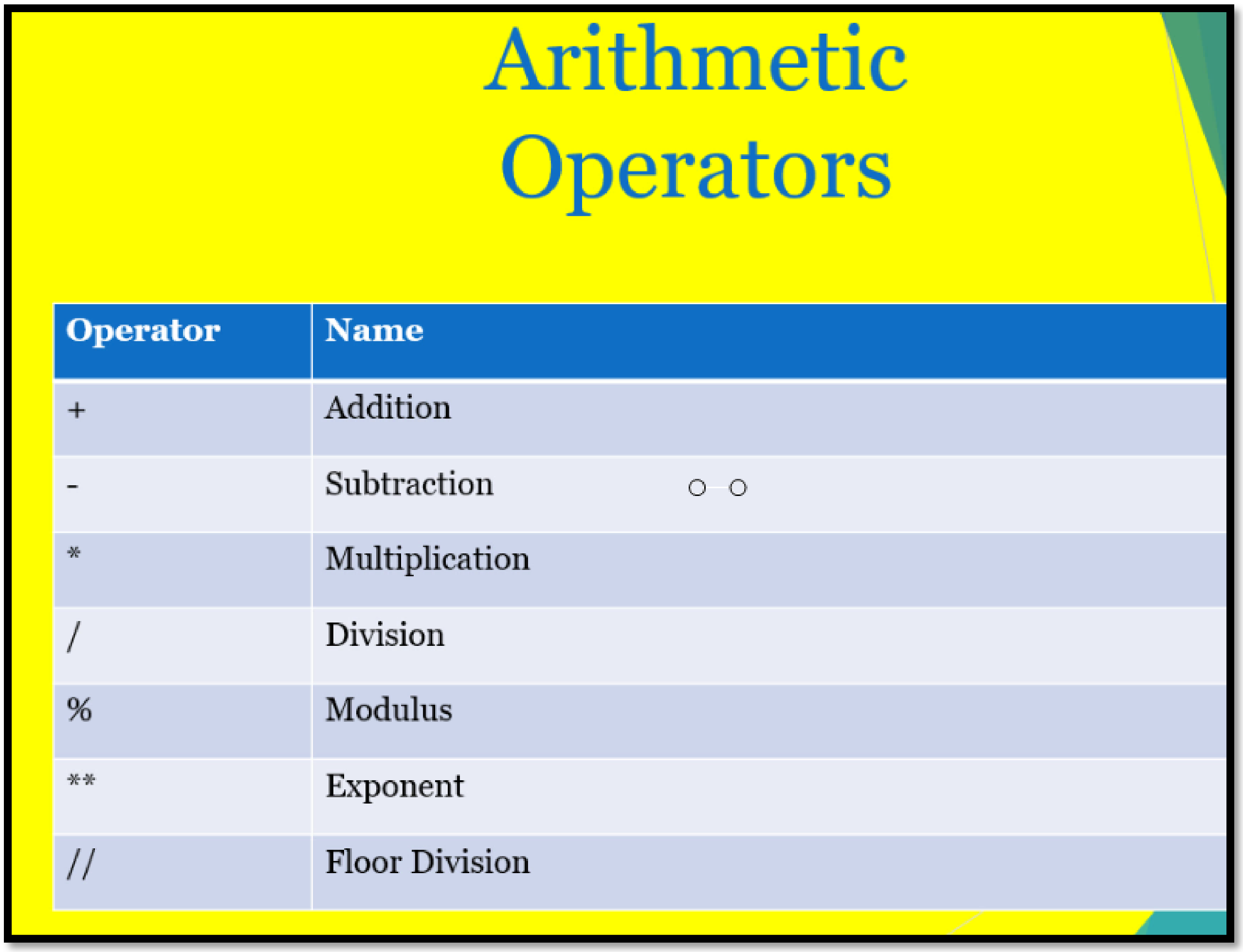
- Assignment – assigns a value to a variable
- Arithmetic – performs math on numeric values
- Comparison – compares 2 values
- Logical – combines conditional statements
- Boolean – expression representing True or False

Assignment Operators

Let’s start with the assignment operator. We have used the assignment operator since naming and creating a variable. It is the equal symbol that assigns a value. For example, course = “Python Step-By-Step”. This statement assigned Python Step-By-Step to course using the equal symbol.

Arithmetic Operators

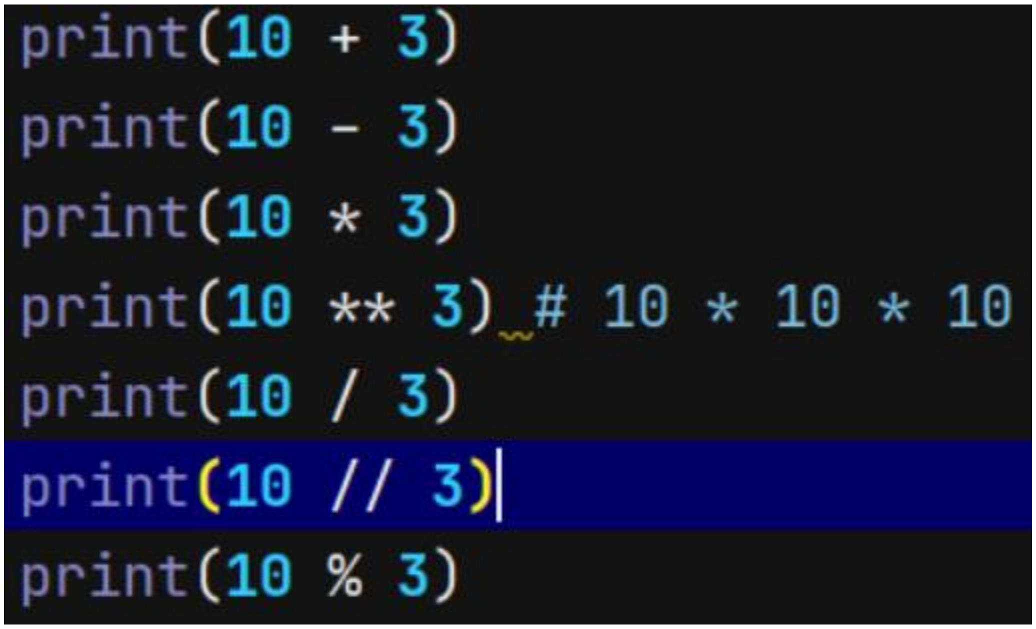
When it comes to an Arithmetic Operator. We have 7 operators: Addition, Subtraction, Multiplication, Division, Modulus, Exponent, and Floor Division.



Arithmetic Operators

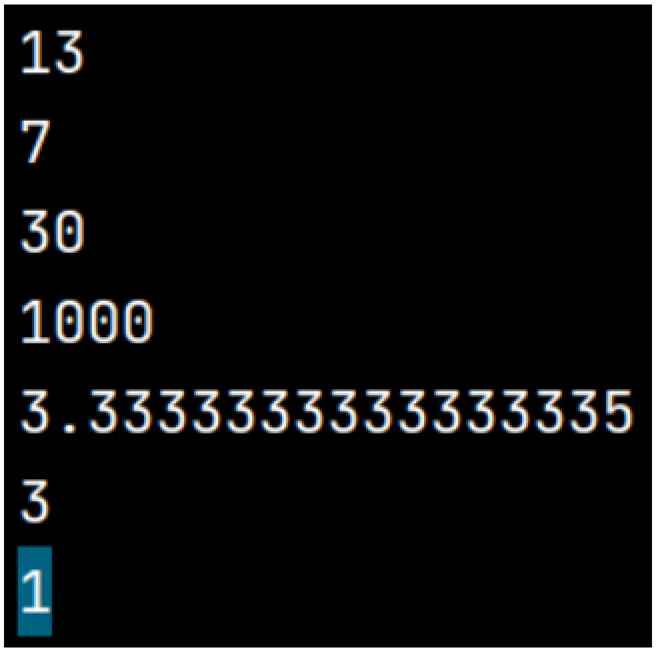
Operator	Name
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
**	Exponent
//	Floor Division

Recall, in Python, there are 2 types of numbers: we have integers and floating point numbers. Integers are whole numbers and Floating point are numbers with a decimal point. The Arithmetic operators work on both types. Let's look at a few operators such as print(10 + 3). We know, the plus symbol represents addition so the answer is 13. The console returns 13. I'm going to copy this statement and paste it 6 times. Subtract (-), Multiply (*), Exponent (**), Divide (/), Floor Division (//), and Modulus (%).



```
print(10 + 3)
print(10 - 3)
print(10 * 3)
print(10 ** 3) # 10 * 10 * 10
print(10 / 3)
print(10 // 3)
print(10 % 3)
```

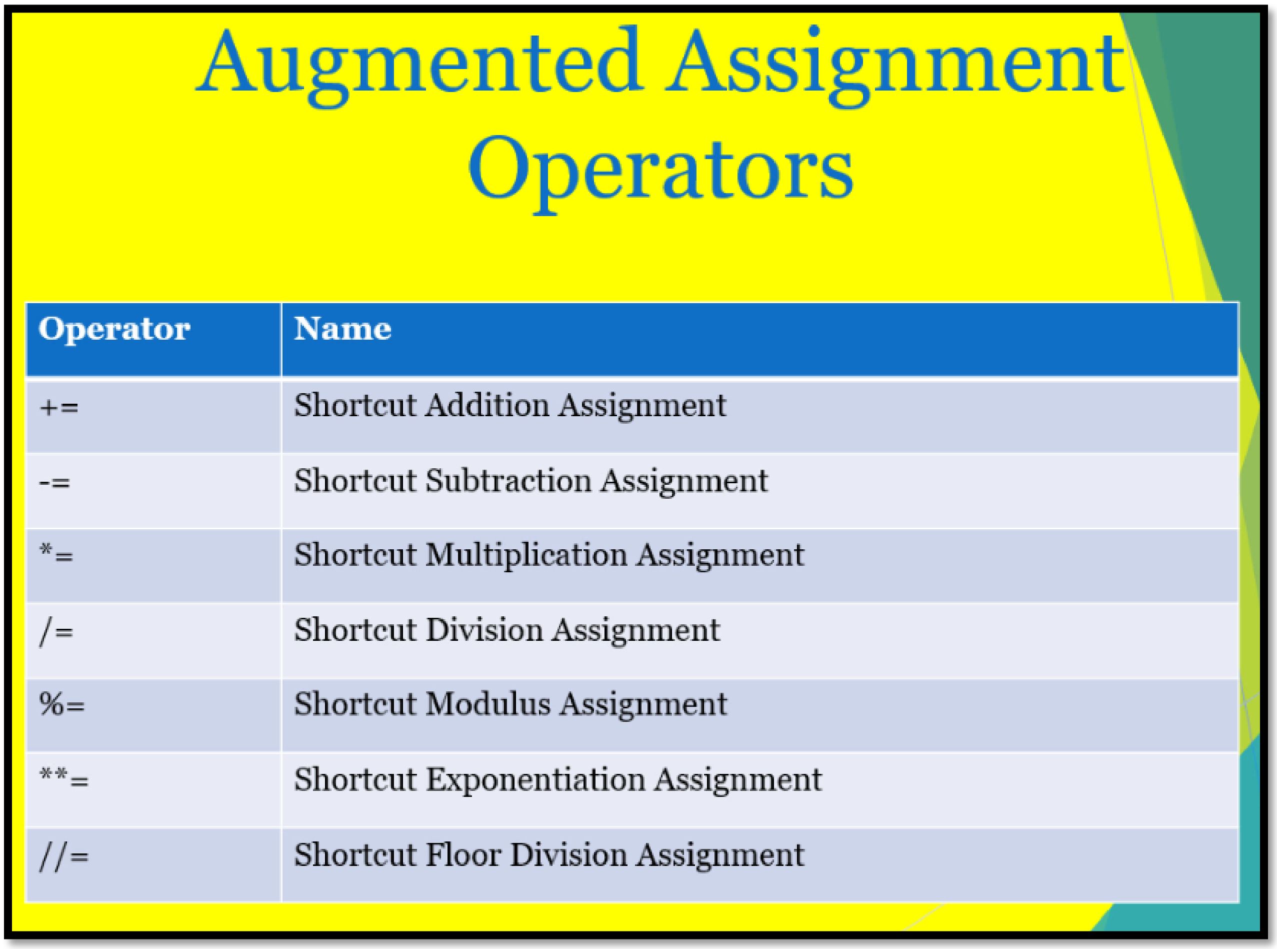
When I run. Subtraction is 7. Multiplication is 30, The exponent raises 10 to the power of 3 then returned 1,000. That's 10 times 10 times 10. We have those 2 Division operators. The normal division operator shows 3.3 with 5 at the end and the floor division operator is also known as an integer division. It only shows the whole number of a division problem. That's why we don't see a decimal so 3 because 10 divided by 3 is 3. It does not include the decimal. The last value is a modulus. That's when we divide one number by another number then return the remainder. The remainder of 10 divided by 3 is 1.



```
13
7
30
1000
3.3333333333333335
3
1
```

Augmented Assignment Operators

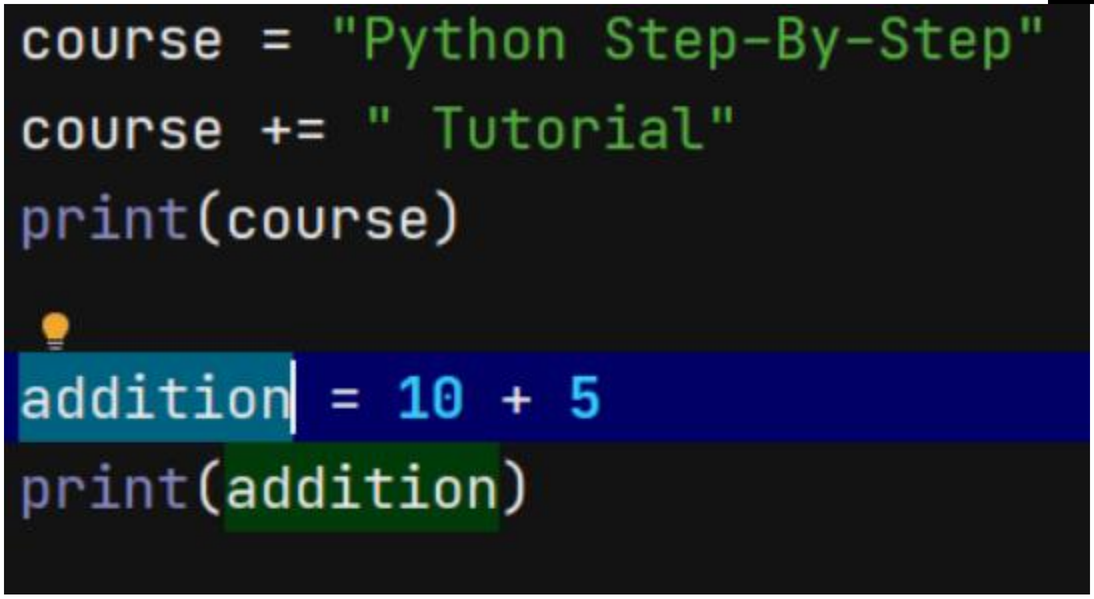
For all of these Arithmetic Operators, we also have Augmented Assignment Operators. An Augmented Assignment Operator is also known as a shortcut operator because it combines 1 Operator with the Assignment Operator. For example, the 1st row shows a plus symbol and an equal symbol. There's a total of 13 but these are the ones we just went over. We can also combine strings using an Augmented Operator.



Augmented Assignment Operators

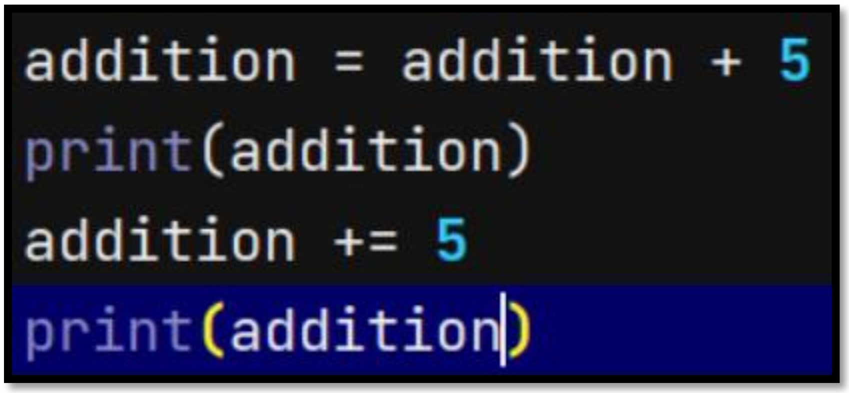
Operator	Name
+=	Shortcut Addition Assignment
-=	Shortcut Subtraction Assignment
*=	Shortcut Multiplication Assignment
/=	Shortcut Division Assignment
%=	Shortcut Modulus Assignment
**=	Shortcut Exponentiation Assignment
//=	Shortcut Floor Division Assignment

In the IDE, we have course equal to Python Step-By-Step. On the next line, I will write course += " Tutorial". When I print(course). We are going to see Python Step-By-Step Tutorial. You see how it combined the first value with the second value. It's the same with addition and the other Arithmetic Operators. We can write addition = 10 + 5. print(addition) The interpreter will add 10 + 5 which is 15 then assign 15 to the addition variable.



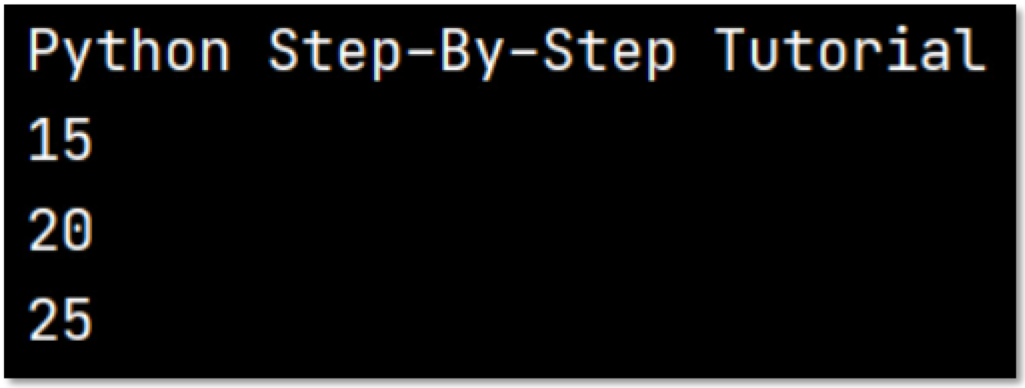
```
course = "Python Step-By-Step"
course += " Tutorial"
print(course)

addition = 10 + 5
print(addition)
```

As expected, the console returns 15. The long way of incrementing this statement by 5 is to write addition = addition + 5. Our shortcut of incrementing by 5 is addition += 5.



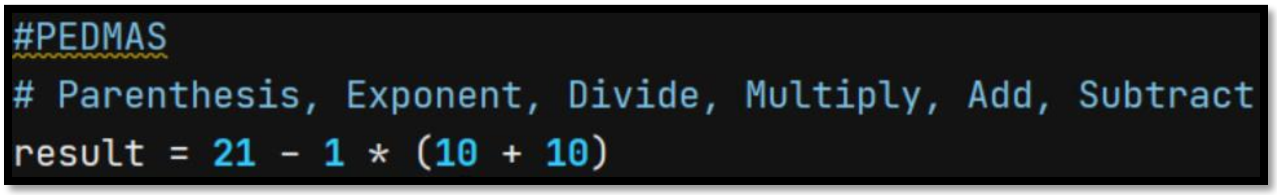
```
addition = addition + 5
print(addition)
addition += 5
print(addition)
```

It's the same process. The 1st value is 15, 2nd value is 20, and the last value is 25 because we are incrementing by 5. print(addition) / print(addition). Run and the console shows 15, 20, and 25



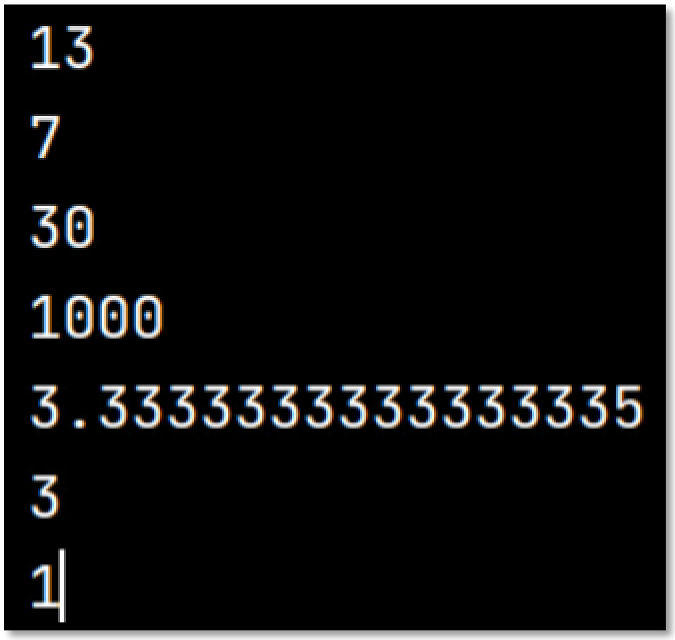
```
Python Step-By-Step Tutorial
15
20
25
```

There's something else I want to show you. It's the Order of Operations also known as Operator Precedence. In Math, it's called #PEDMAS which stands for #Parenthesis, Exponent, Divide, Multiply, Add, Subtract. It provides the order of how our program will execute the Arithmetic Operators. For example, let's say result = has a value of 21 – 1 * (10 + 10). Looking at this statement, a person might think we solve this problem starting from left to right. But using PEDMAS, we begin with the Parenthesis of 10 + 10 equal to 20 then multiply by 1 which remains at 20 because 20 * 1 is 20. Now, we have 21 – 20 which is 1.



```
#PEDMAS
# Parenthesis, Exponent, Divide, Multiply, Add, Subtract
result = 21 - 1 * (10 + 10)
```

Run and the console shows 1. That's it for Assignment & Arithmetic Operators.



```
13
7
30
1000
3.3333333333333335
3
1
```

Contact Info

- ✓ Email Rex.Jones@Test4Success.org
- ✓ YouTube https://www.youtube.com/c/RexJonesII/videos
- ✓ Facebook https://facebook.com/JonesRexII
- ✓ Twitter https://twitter.com/RexJonesII
- ✓ GitHub https://github.com/RexJonesII/Free-Videos

- ✓ LinkedIn https://www.linkedin.com/in/rexjones34/