Floor Division in Python

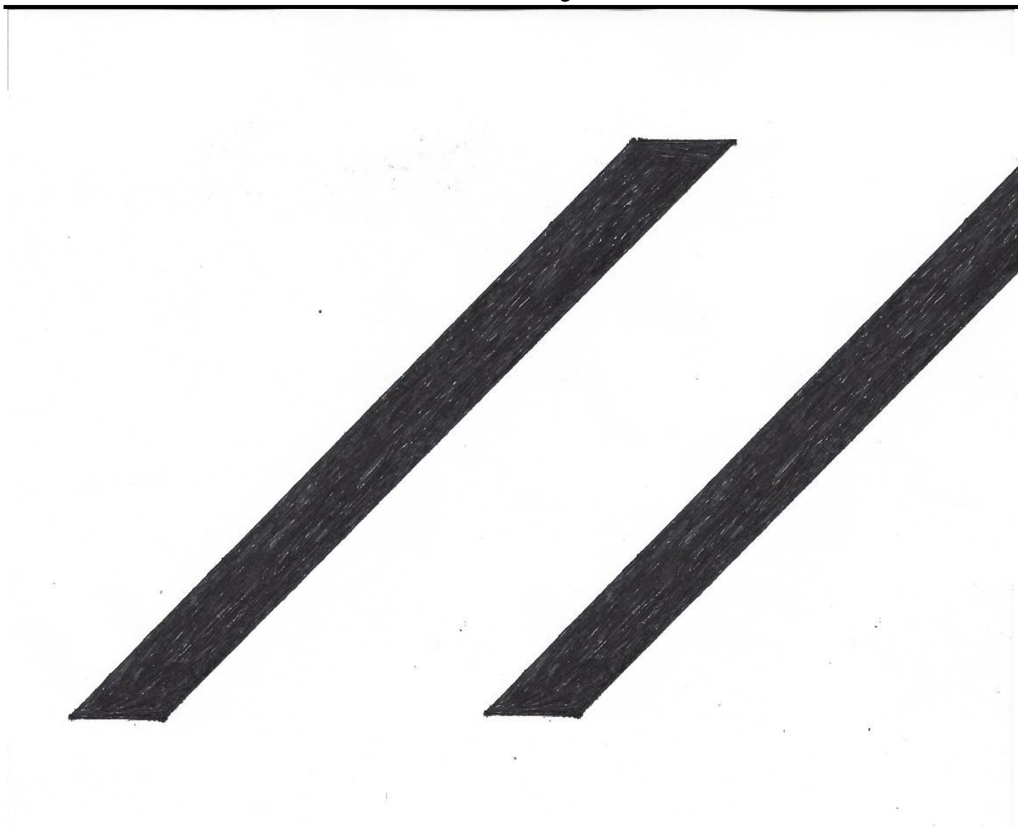


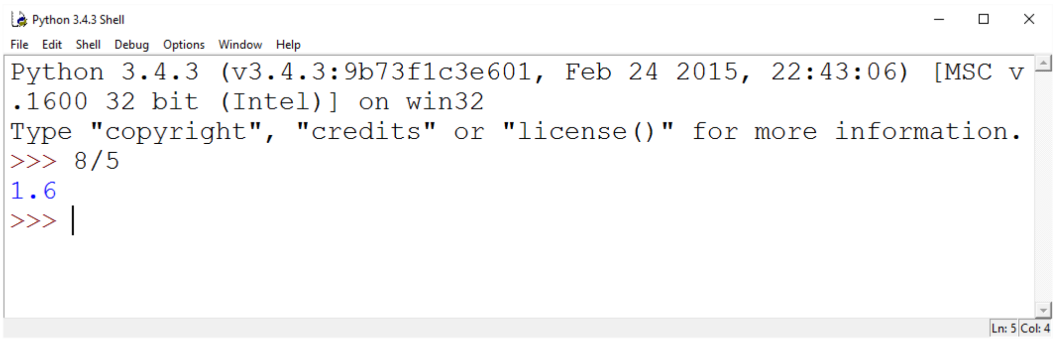
Figure 1: The Floor-Division operator. In Python, the Floor-Division operator consists of two forward slashes. The Floor-Division operator is an example of a binary operator, as it takes two operands: the dividend and the divisor.

With floor division, one number, the dividend, is divided by another number, the divisor, and the result, or quotient – whatever it may happen to be – will be a rounded-down integer value.

Let us consider the Python Equation:

```
>>>8/5
1.6
>>>
```

The number, 8, the dividend, is divided by the divisor, 5, and a floating-point number, 1.6, is then returned as a quotient.



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 8/5
1.6
>>> |
```

Figure 2: When we divide 8 by 5 using the division operator, / , then a floating-point number, 1.6, is returned as a quotient.

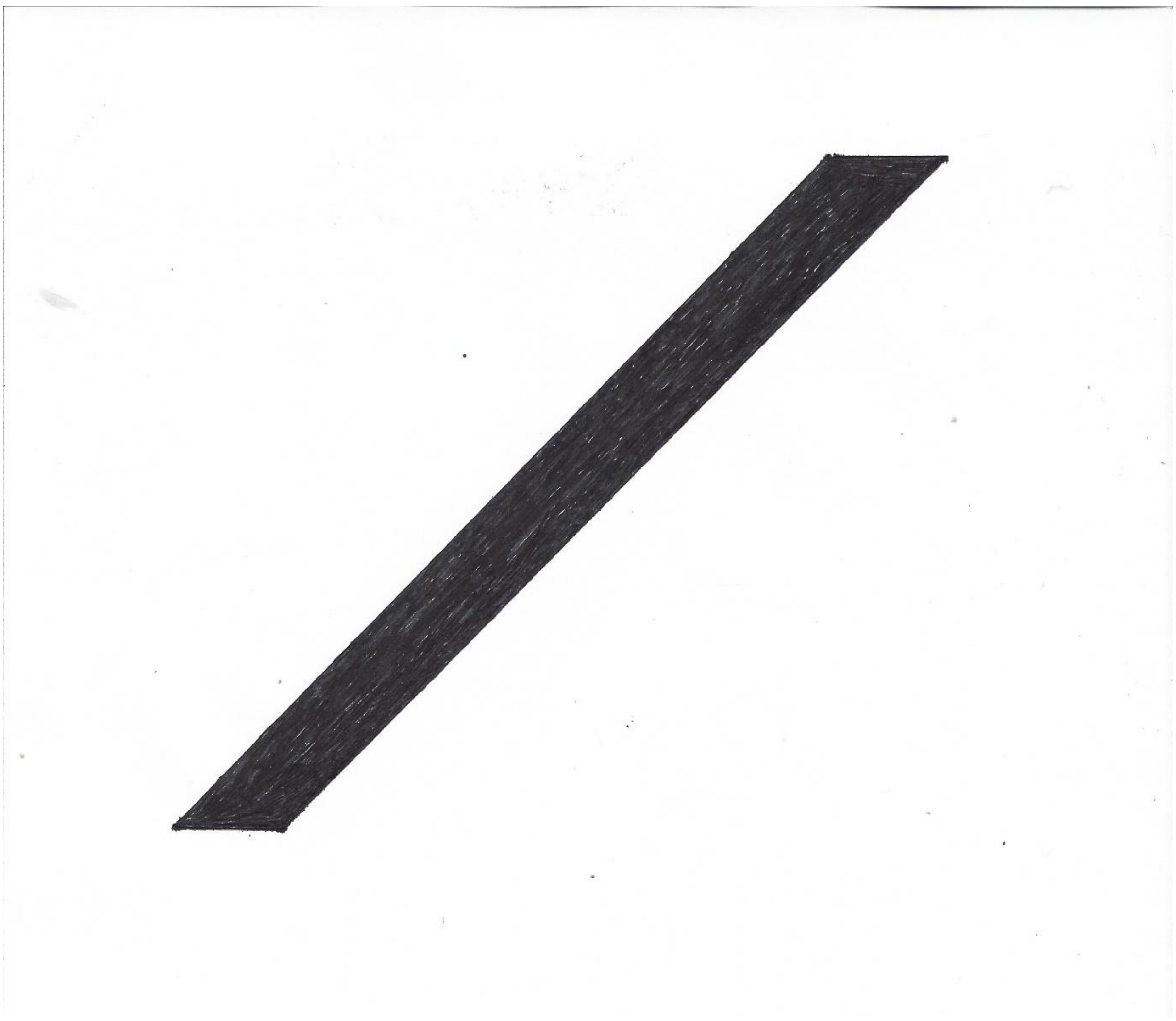
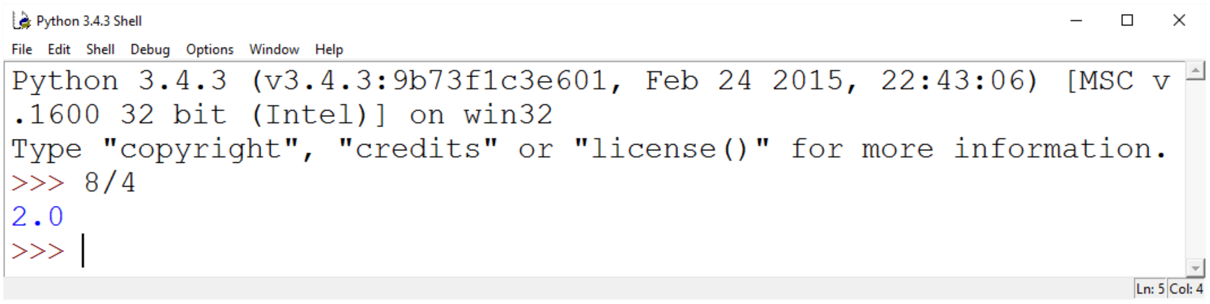


Figure 3: This is our Division operator. When we employ this binary operator, a floating-point number will be returned.

Whenever we employ a Division operator in Python, then a floating point number will always be returned as a quotient, even if the quotient has no significant fractional component.



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 8/4
2.0
>>> 
```

Figure 4: Whenever we divide the dividend, 8, by the divisor, 4, then the quotient, 2.0, is still returned as a floating-point number despite its not having any significant fractional component.

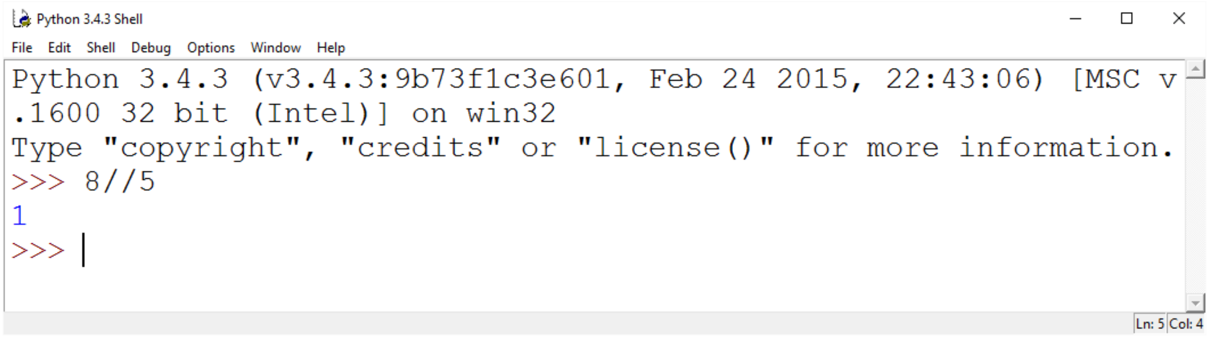
Let us, again, consider the Python equation:

```
>>>8/5
1.6
>>>
```

, but let us do things a little differently:

```
>>>8//5
1
>>>
```

In the above example, we have now employed the floor-division operator. The floor-division operator will *always* return an integer value, if the 2 operands that it takes be integers.



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 8//5
1
>>> 
```

Figure 5: When we divide the dividend, 8, by the divisor, 5, we get the quotient, 1, rendered as an integer.

Let us consider 8 divided by 2 in ordinary arithmetic for a moment:

$$8 \div 5 = 1.6$$

In the above example, we divide an integer by an integer and we obtain a real number as a result, or quotient.

If we wanted a less precise answer, then it would be customary to see:

$$8 \div 5 \approx 2$$

In normal arithmetic, it would be customary to round:

$$1.6$$

up to:

$$2$$

.

However, in floor division, floating point numbers such as:

```
>>>1.6
1.6
>>>
```

are always rounded *down* to the value of its integral component.

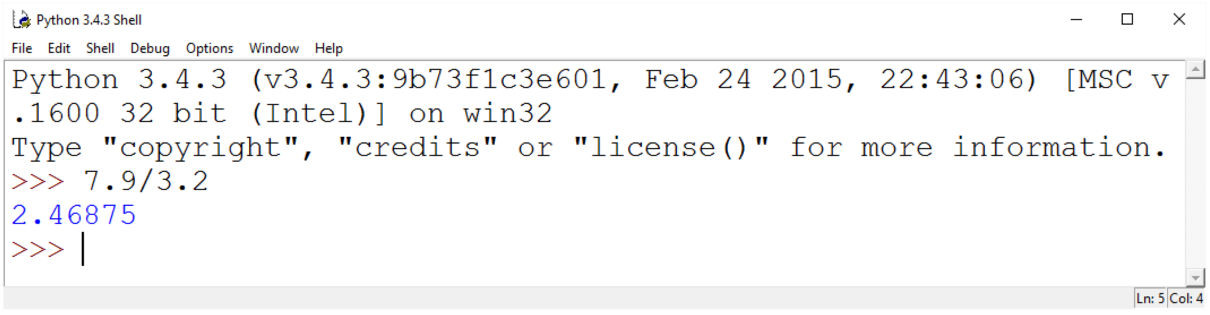
So, in Python, the *floor value* of:

```
>>>1.6
1.6
>>>
```

would be:

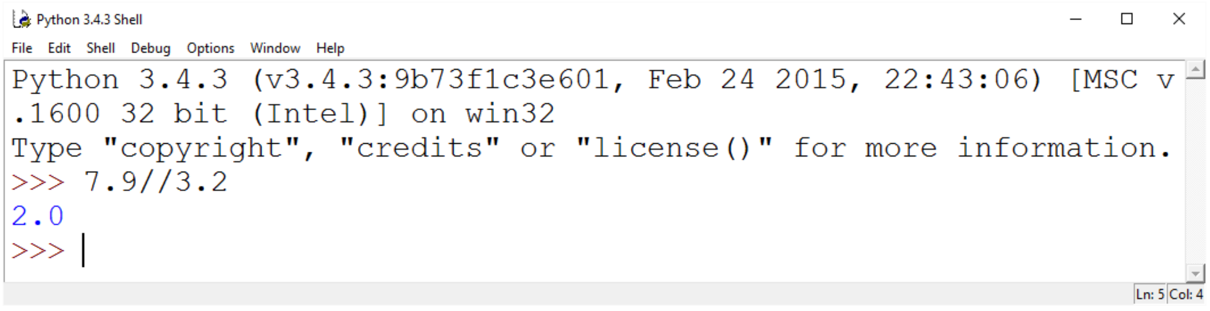
```
>>>2
2
>>>
```

The floor-division operator will always return an integer as a quotient, unless floating-point numbers be employed as operands.



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 7.9/3.2
2.46875
>>> 
```

Figure 6: When we divide 7.9 by 3.2 in conventional division, we obtain the floating-point quotient, 2.46875

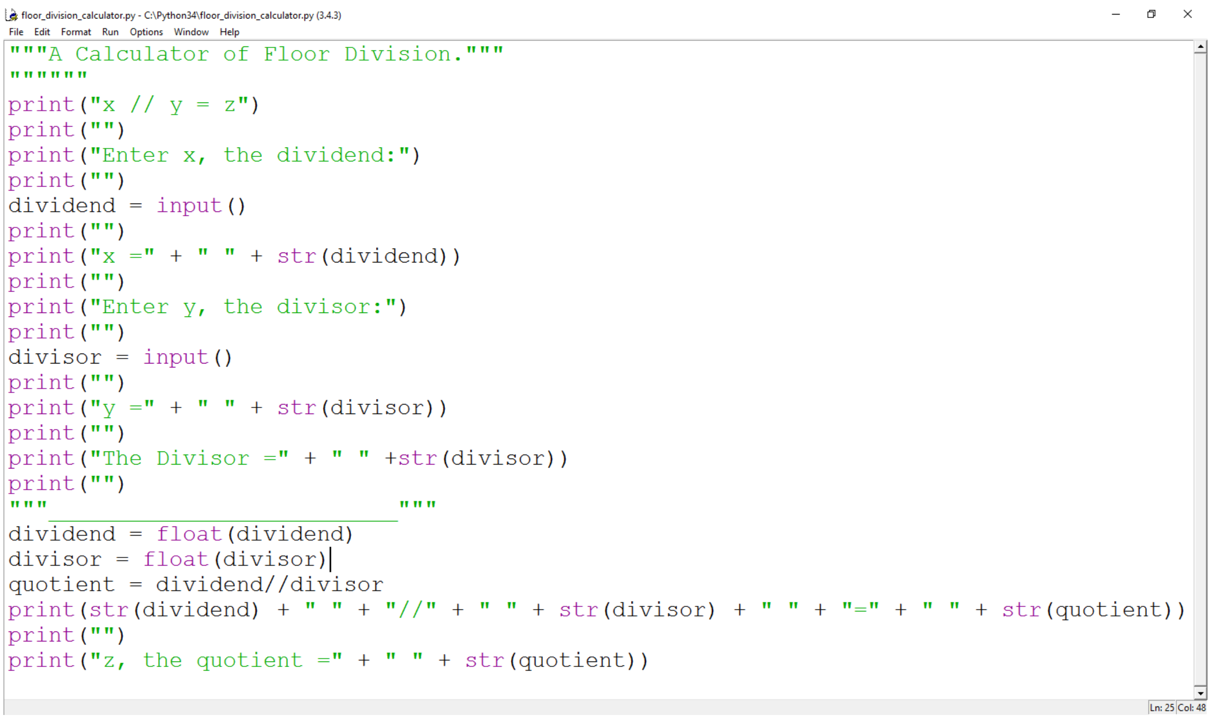


```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 7.9//3.2
2.0
>>> 
```

Figure 7: When we divide 7.9 by 3.2 in floor division, we still obtain a floating-point quotient, 2.0, but it does not have a significant fractional component.

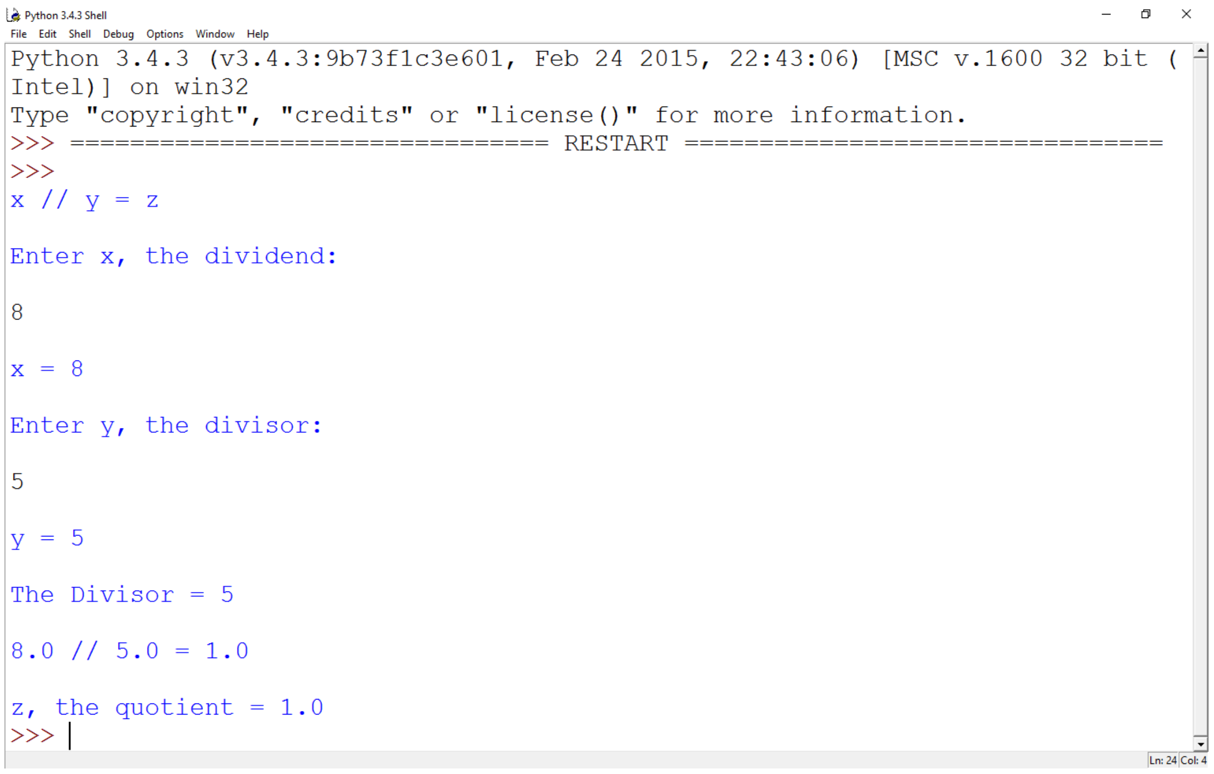
Programming a Floor-Division Calculator in Python:

In the following section, we shall program a simple floor-division calculator in Python:


floor_division_calculator.py - C:\Python34\floor_division_calculator.py (3.4.3)
File Edit Format Run Options Window Help
Ln: 25 Col: 48

```python
"""A Calculator of Floor Division."""
""""""
print("x // y = z")
print("")
print("Enter x, the dividend:")
print("")
dividend = input()
print("")
print("x =" + " " + str(dividend))
print("")
print("Enter y, the divisor:")
print("")
divisor = input()
print("")
print("y =" + " " + str(divisor))
print("")
print("The Divisor =" + " " +str(divisor))
print("")
"""________________________"""
dividend = float(dividend)
divisor = float(divisor)
quotient = dividend//divisor
print(str(dividend) + " " + "//" + " " + str(divisor) + " " + "=" + " " + str(quotient))
print("")
print("z, the quotient =" + " " + str(quotient))
```

Figure 8: A simple floor-division calculator programmed in Python. This program requests that the user input two numbers. The program then takes these inputs; divides the dividend by the divisor; and then returns a rounded-down quotient.



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
x // y = z

Enter x, the dividend:

8

x = 8

Enter y, the divisor:

5

y = 5

The Divisor = 5

8.0 // 5.0 = 1.0

z, the quotient = 1.0
>>> 
```

Figure 9: What the previous program, depicted in **Figure 8**, outputs when run.