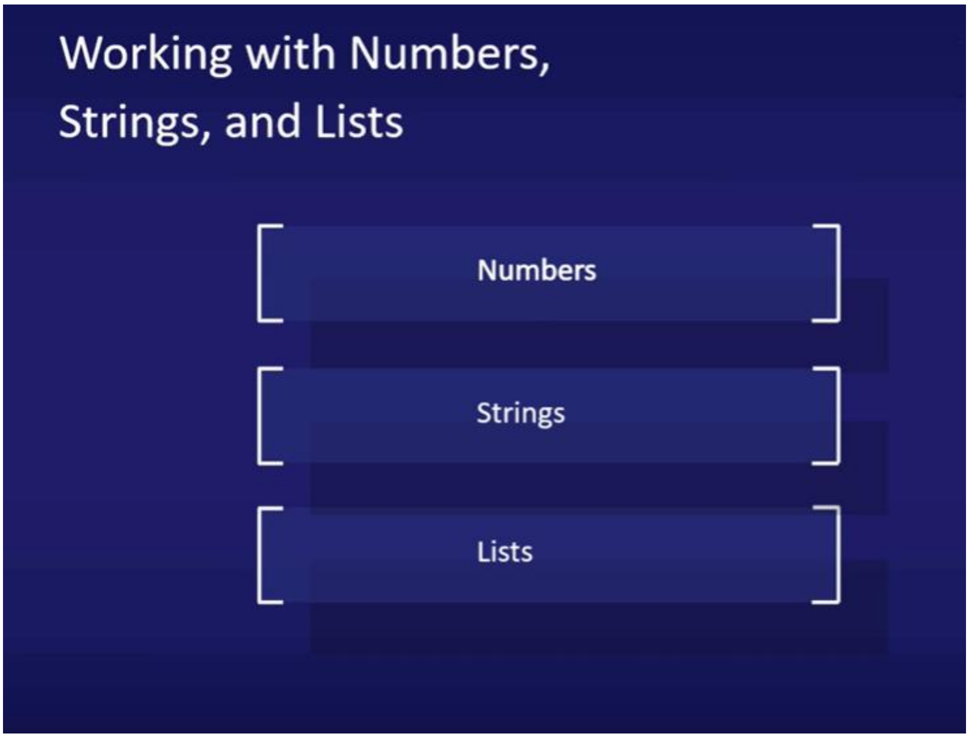
Python Scripting:

Variables, Data Types and Operators:



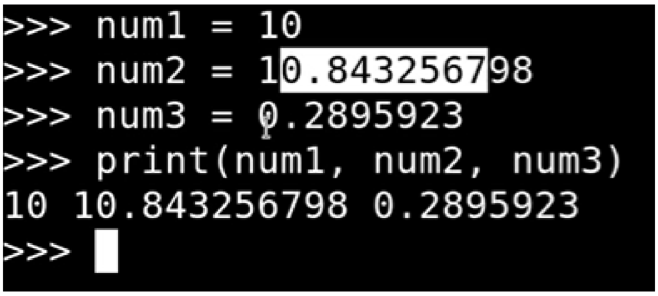
Working with Numbers, Strings, and Lists

Numbers

Strings

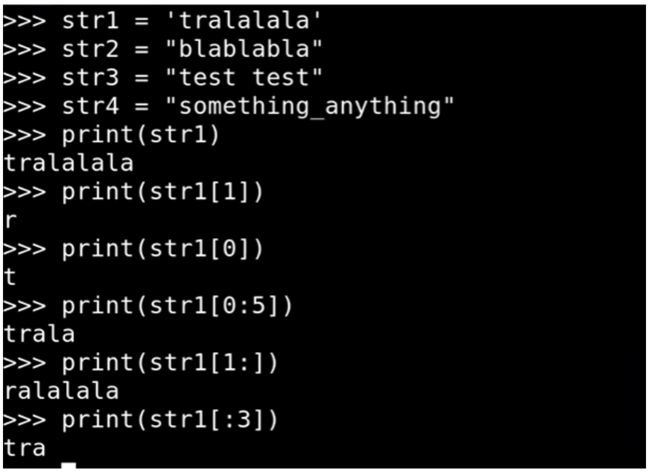
Lists

Numbers:



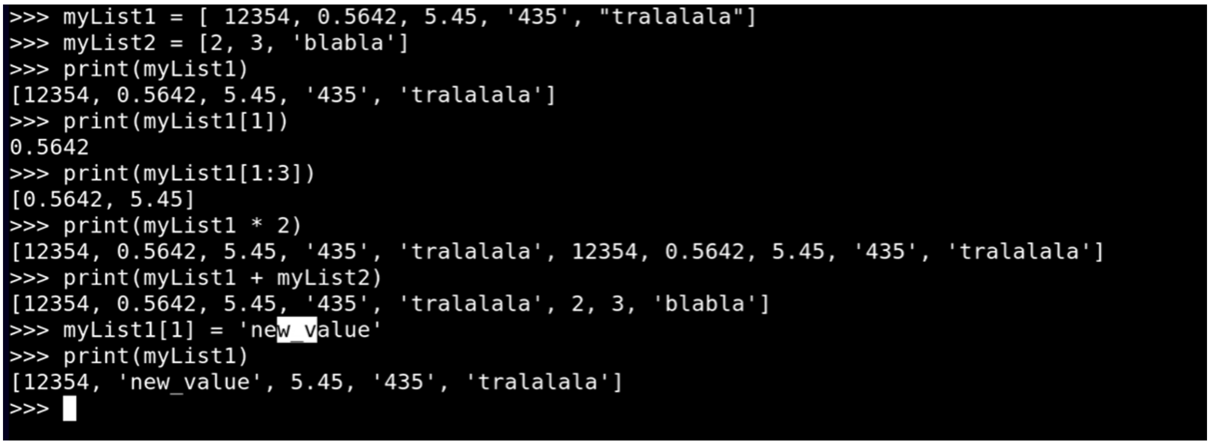
```
>>> num1 = 10
>>> num2 = 10.843256798
>>> num3 = 0.2895923
>>> print(num1, num2, num3)
10 10.843256798 0.2895923
>>> 
```

Strings:

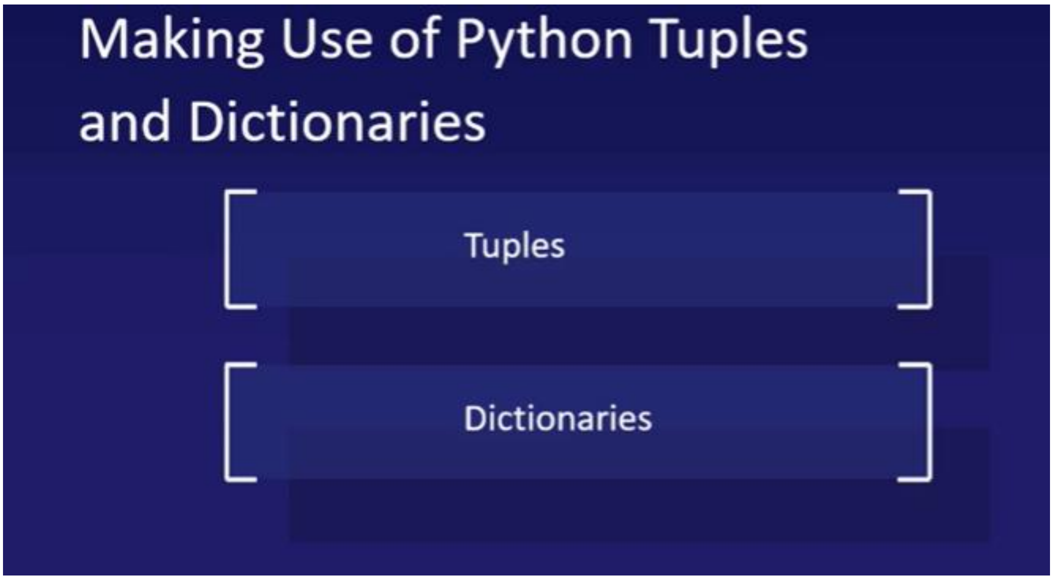


```
>>> str1 = 'tralalala'
>>> str2 = "blablabla"
>>> str3 = "test test"
>>> str4 = "something_anything"
>>> print(str1)
tralalala
>>> print(str1[1])
r
>>> print(str1[0])
t
>>> print(str1[0:5])
trala
>>> print(str1[1:])
ralalala
>>> print(str1[:3])
tra
```

3. Lists:



```
>>> myList1 = [ 12354, 0.5642, 5.45, '435', "tralalala"]
>>> myList2 = [2, 3, 'blabla']
>>> print(myList1)
[12354, 0.5642, 5.45, '435', 'tralalala']
>>> print(myList1[1])
0.5642
>>> print(myList1[1:3])
[0.5642, 5.45]
>>> print(myList1 * 2)
[12354, 0.5642, 5.45, '435', 'tralalala', 12354, 0.5642, 5.45, '435', 'tralalala']
>>> print(myList1 + myList2)
[12354, 0.5642, 5.45, '435', 'tralalala', 2, 3, 'blabla']
>>> myList1[1] = 'new_value'
>>> print(myList1)
[12354, 'new_value', 5.45, '435', 'tralalala']
>>> 
```

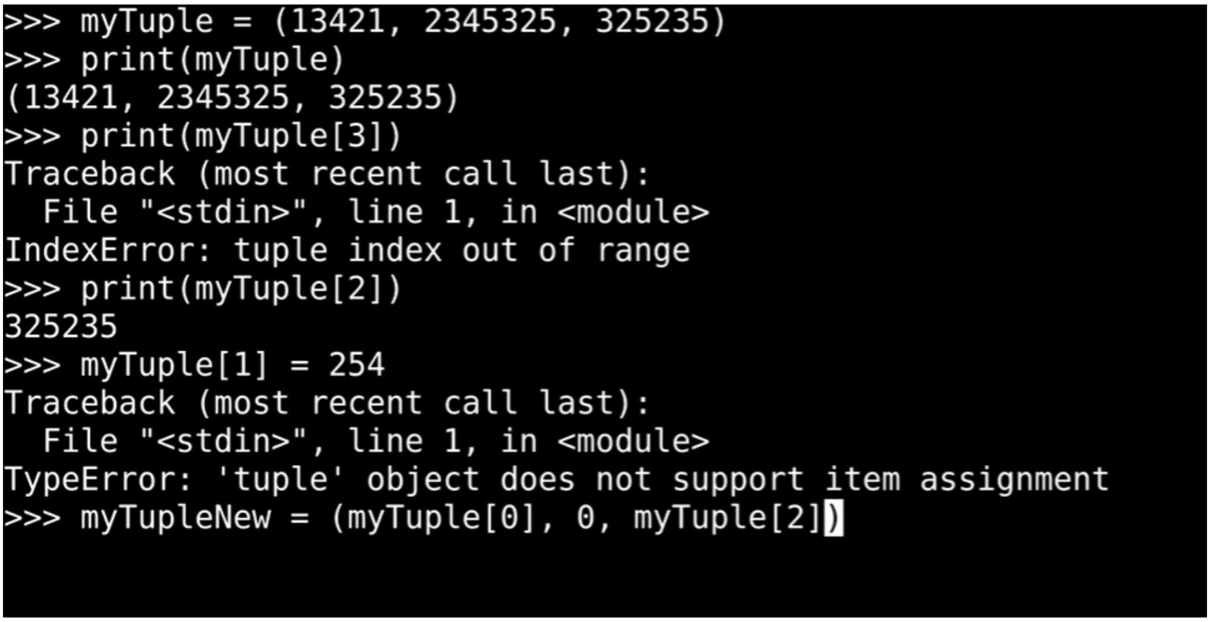


Making Use of Python Tuples and Dictionaries

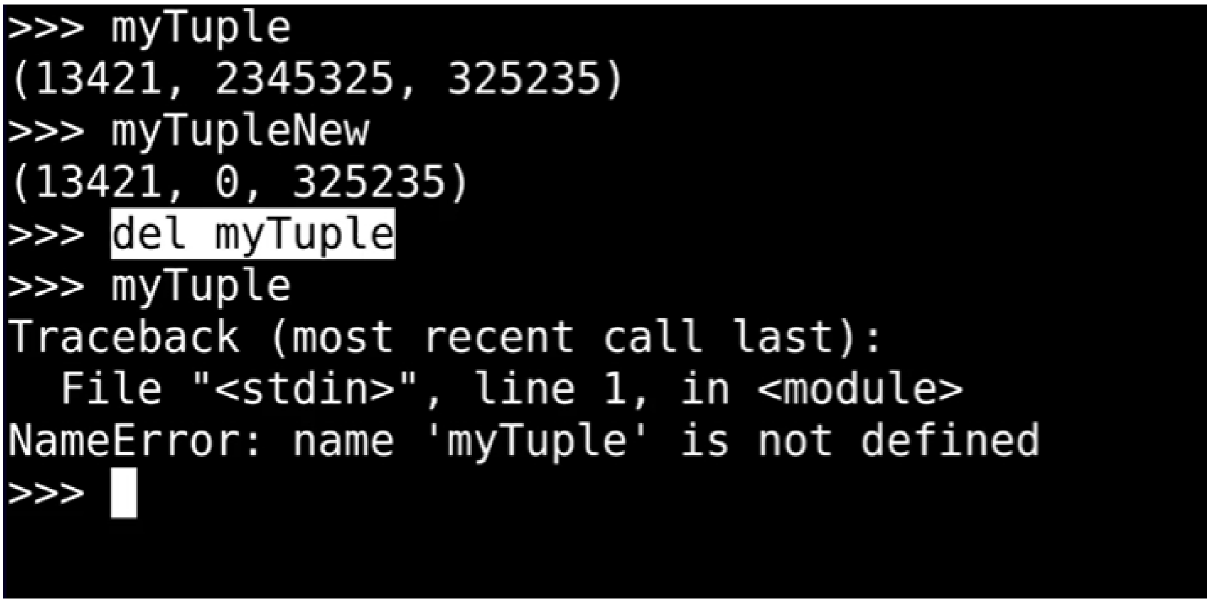
Tuples

Dictionaries

Tuples:

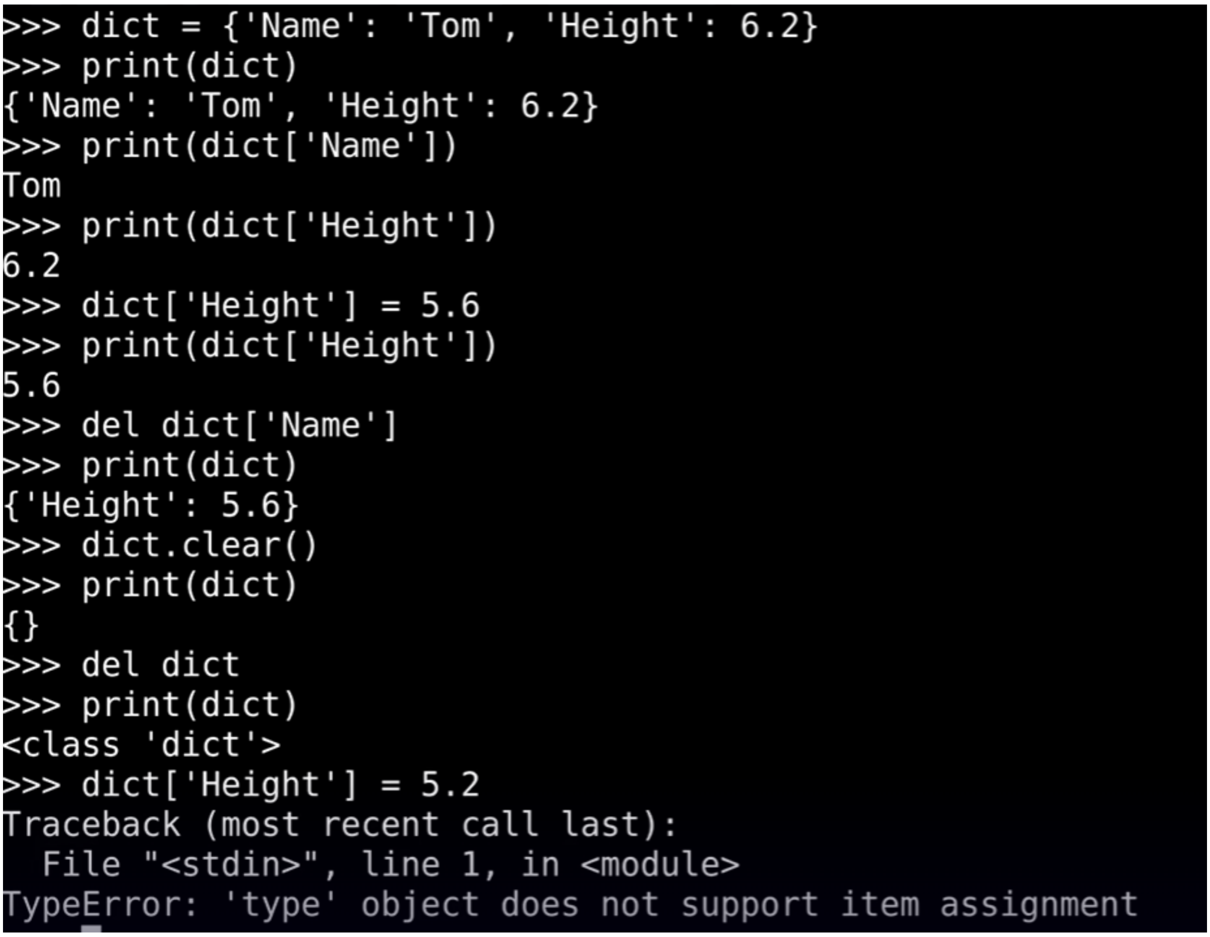


```
>>> myTuple = (13421, 2345325, 325235)
>>> print(myTuple)
(13421, 2345325, 325235)
>>> print(myTuple[3])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: tuple index out of range
>>> print(myTuple[2])
325235
>>> myTuple[1] = 254
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> myTupleNew = (myTuple[0], 0, myTuple[2])
```



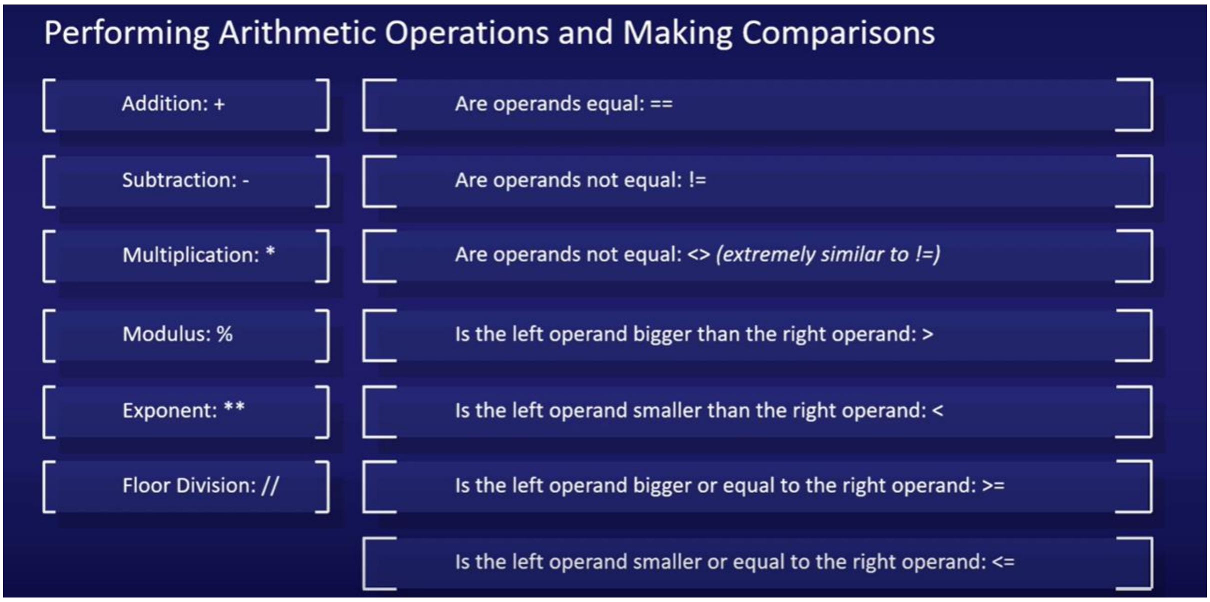
```
>>> myTuple
(13421, 2345325, 325235)
>>> myTupleNew
(13421, 0, 325235)
>>> del myTuple
>>> myTuple
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'myTuple' is not defined
>>> 
```

Dictionaries:



```
>>> dict = {'Name': 'Tom', 'Height': 6.2}
>>> print(dict)
{'Name': 'Tom', 'Height': 6.2}
>>> print(dict['Name'])
Tom
>>> print(dict['Height'])
6.2
>>> dict['Height'] = 5.6
>>> print(dict['Height'])
5.6
>>> del dict['Name']
>>> print(dict)
{'Height': 5.6}
>>> dict.clear()
>>> print(dict)
{}
>>> del dict
>>> print(dict)
<class 'dict'>
>>> dict['Height'] = 5.2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'type' object does not support item assignment
```

Arithmetic Operations:



Performing Arithmetic Operations and Making Comparisons

Addition: +

Subtraction: -

Multiplication: *

Modulus: %

Exponent: **

Floor Division: //

Are operands equal: ==

Are operands not equal: !=

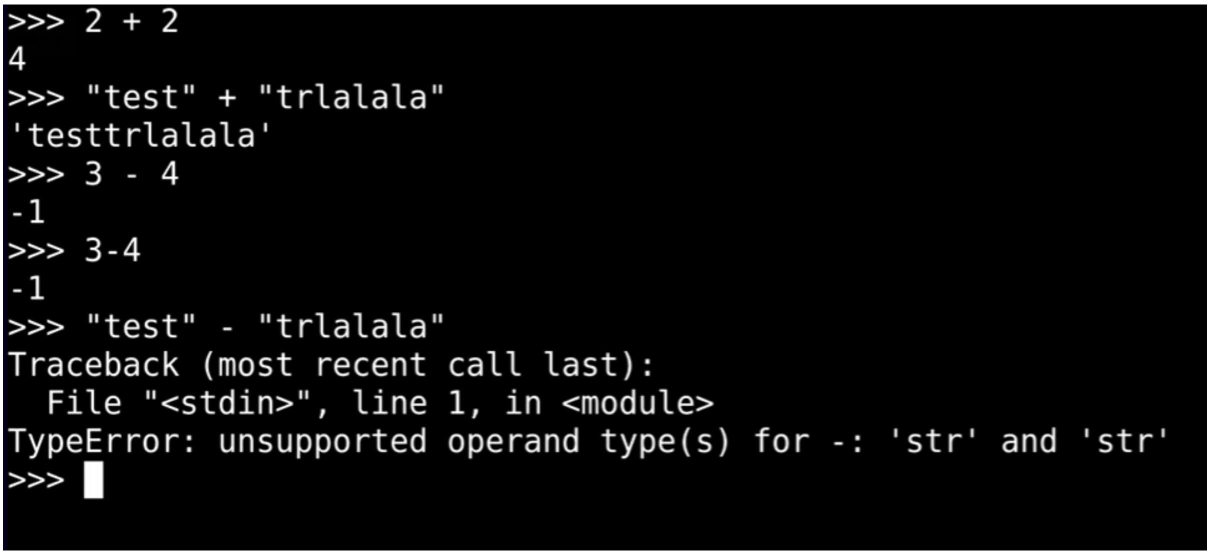
Are operands not equal: <> *(extremely similar to !=)*

Is the left operand bigger than the right operand: >

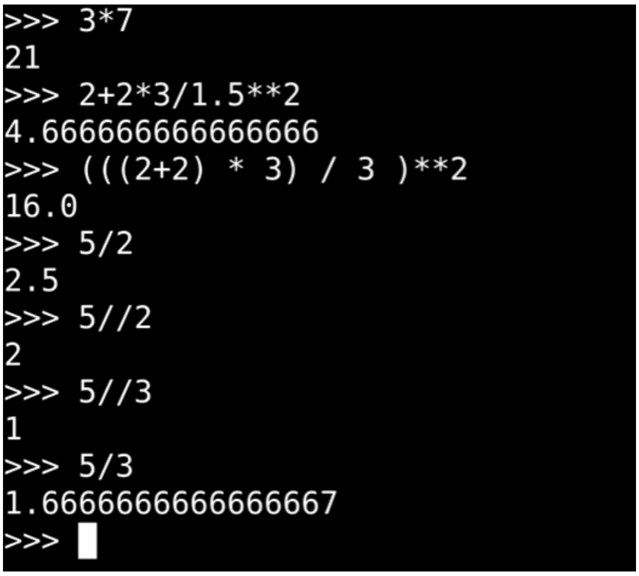
Is the left operand smaller than the right operand: <

Is the left operand bigger or equal to the right operand: >=

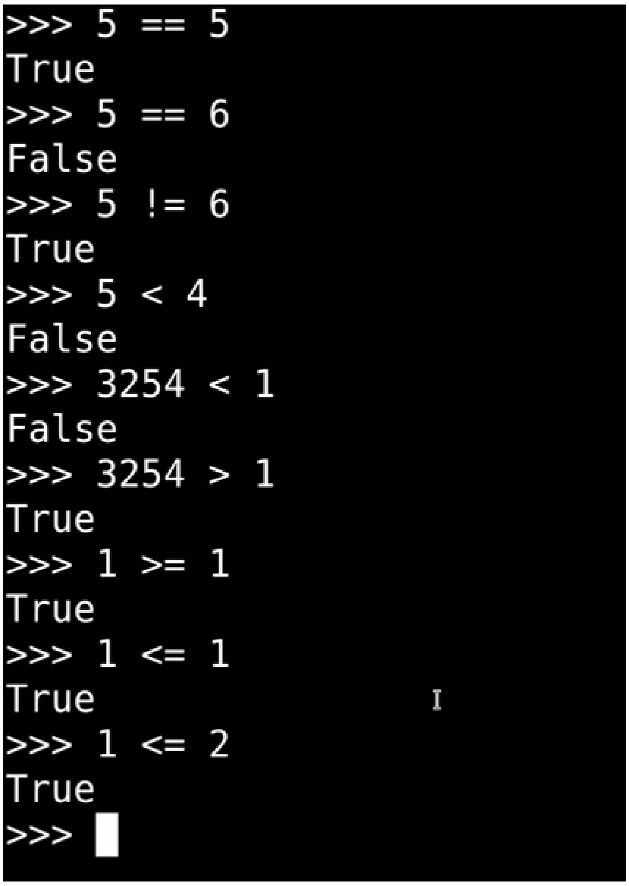
Is the left operand smaller or equal to the right operand: <=



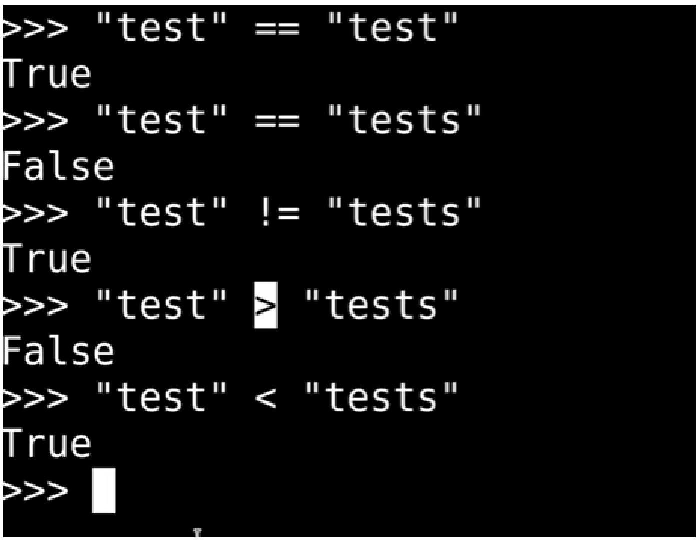
```
>>> 2 + 2
4
>>> "test" + "trlalala"
'testtrlalala'
>>> 3 - 4
-1
>>> 3-4
-1
>>> "test" - "trlalala"
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for -: 'str' and 'str'
>>> 
```



```
>>> 3*7
21
>>> 2+2*3/1.5**2
4.666666666666666
>>> (((2+2) * 3) / 3 )**2
16.0
>>> 5/2
2.5
>>> 5//2
2
>>> 5//3
1
>>> 5/3
1.6666666666666667
>>> 
```



```
>>> 5 == 5
True
>>> 5 == 6
False
>>> 5 != 6
True
>>> 5 < 4
False
>>> 3254 < 1
False
>>> 3254 > 1
True
>>> 1 >= 1
True
>>> 1 <= 1
True
>>> 1 <= 2
True
>>> 
```



```
>>> "test" == "test"
True
>>> "test" == "tests"
False
>>> "test" != "tests"
True
>>> "test" > "tests"
False
>>> "test" < "tests"
True
>>> 
```