REPL

The python interpreter, when run in interpreter mode, yields a REPL.

Numbers

Here is some python code involving integers, real numbers, and complex numbers and their operations.

```
>>> x = int(45)
>>> y = 9
>>> x / y
5.0
>>> x // y
5
>>> z = float(9)
>>> x / z
5.0
>>> z * x
405.0
>>> y * x
405
>>> u = 9.0
>>> x / u
5.0
>>> y**2
81
>>> y % 3
0
>>> y % 2
1
>>> y % 5
4
>>> a = complex(1,0)
>>> b = complex(0,1)
>>> a+b
(1+1j)
>>> a*b
1j
>>> 1j**2
(-1+0j)
>>> 1j**2 + 1.0
0j
```

Strings

```
>>> x = "Brent's sister's husband's brother-in-law is a great guy."
>>> x
"Brent's sister's husband's brother-in-law is a great guy."
>>> y = 'Brent says, "Good thing my brother-in-law is an only child."'
>>> y
'Brent says, "Good thing my brother-in-law is an only child."'
>>> z = x + " " + y
>>> z
'Brent\'s sister\'s husband\'s brother-in-law is a great guy. Brent says, "Good thing my brother-in-law is an only child."'
>>> print(x)
Brent's sister's husband's brother-in-law is a great guy.
>>> print(y)
Brent says, "Good thing my brother-in-law is an only child."
>>> print(z)
Brent's sister's husband's brother-in-law is a great guy. Brent says, "Good thing my brother-in-law is an only child."
>>> a = "a newline\ncharacter"
>>> print(a)
a newline
character
>>> a = r'a newline\ncharacter'
>>> a
'a newline\\ncharacter'
>>> print(a)
a newline\ncharacter
```

Consider the following interaction on the Python interpreter. What is `x`?

```
>>> print(x)
She said, "Brent's favorite character is \n."
He said, "I know."
```

Python

Let's write a program called `sum.py` that takes two arguments from the command line and prints out their sum.

```
import sys

x = sys.argv[1]
y = sys.argv[2]

print("The sum of " + x + " and " + y + " is " + (x+y))
```

First some explanation. The module `sys` gives us access to a variable called `argv`, which is a vector of strings that appear on the command line. `sys.argv[0]` is the name of the script. `sys.argv[1]` is the first argument, `sys.argv[2]` is the second argument, and so on. Let's run this script in script mode.

```
$ python3 sum.py 5 6
The sum of 5 and 6 is 56
```

Um, that's not right. What's wrong? The arguments are strings of characters, not numbers.

```
import sys

x = int(sys.argv[1])
y = int(sys.argv[2])

print("The sum of " + str(x) + " and " + str(y) + " is " + str(x+y))
```