Exercise 5: Introduction to Python

In this exercise, we will go through some of the basics of Python using Jupyter Notebook, the Python interpreter, and scripts.

Jupyter Notebook

We will use Jupyter Notebook to demonstrate some of the basics of Python programming. Jupyter Notebook provides an interactive user friendly way of writing and testing snippets of code, but for more complex programs, scripts (described below) are much more efficient.

The `print` function

For our first code, we will have Python output a message using the print function (to execute code in the code boxes, type shift + return):

In [4]:
```
print("Hello, world!")
```

```
Hello, world!
```

Python is very particular about syntax. Try removing the parentheses:

In [5]:
```
print "Hello, world!"
```

```
  File "<ipython-input-5-d07e2790dcc1>", line 1
    print "Hello, world!"
                        ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean prin
t("Hello, world!")?
```

Try removing the quotation marks:

In [6]:
```
print(Hello, world!)
```

```
  File "<ipython-input-6-d14e6f3d0ea2>", line 1
    print(Hello, world!)
                       ^
SyntaxError: invalid syntax
```

Try changing the double quotes to single quotes:

In [7]:
```
print('Hello, world!')
```

```
Hello, world!
```

If we want the message to span multiple lines, we can use triple quotes:

The Python interpreter

Open a terminal and type `python --version` at the prompt (`$`). If your default version of python is 2.x, and you've installed python version 3.x, you can invoke it by typing `python3` .

At the prompt (>>>), use the print function to print the phrase "I love Python!". To exit the interpreter, use ctrl+d.

Python scripts

The Jupyter notebook and the Python intepretor are great for writing and testing snippets of code, but when we want to actually create a Python program we write the instructions in a file, often called a script, using text editing software (e.g. gedit, TextWrangler, and Notepad++).

Open a text editor and create a new document. Save the document as `first_script.py` in a new folder called `Scripts` or some other descriptive name. The `.py` extension is the conventional way of designating a file as a python script but it is not necessary to execute the script for Unix-based operating systems.

Everything in the file will be intepreted as code, unless its commented out with a `#` .

Write a Python script that outputs the message "My first Python script!". The text editing software you use should color the code to make it easier to read. Adding the `.py` extension should be sufficient to trigger syntax coloring within the file.

To execute the script from the command line, type `python` (or `python3` , if Python v2 is the default) followed by the name of the script (e.g. `python3 first_script.py`).

Math

Python does math in a fairly intuitive way. For example, what is 1 plus 1?

In [8]:
```
1+1
```

Out[8]:
```
2
```

What is 1 divided by 2?

In [9]:
```
1/2
```

Out[9]:
```
0.5
```

What is 2 times 3?

In [10]:
```
2*3
```

Out[10]:
```
6
```

What is 2 cubed ('to the power of' uses the syntax `**`)?

In [11]:
```
2**3
```

Out[11]:
```
8
```

Try some more complex math. Does Python follow conventional rules for presedence of mathamatical operations?

In [12]:
```
2-3*3
```

Out[12]:
```
-7
```

In math, we often work with variables, such as X and Y. Try creating a variable `x` and assign a value to it, just like you would if you were doing an algebra problem.

In [13]:
```
x = 5
```

Now use the variable in an equation.

In [14]:
```
x+5
```

Out[14]:
```
10
```

Try returning the value of the variable using the `print` function.

In [15]:
```
print("x")
```

```
x
```

If you followed the syntax used for the the `print` function in the earlier example, you probably got as output exactly what you entered. What happens if you remove the quotation marks?

In [16]:
```
print(x)
```

```
5
```

Try returning the value of a math operation using the `print` function. Test what happens when you include or exclude quotation marks.

In [17]:
```
print("5+5")
```

```
5+5
```

In [18]:
```
print(5+5)
```

```
10
```

In [19]:
```
5+5
```

Out[19]:
```
10
```

Independent Exercise 5

Write a script called `square_root.py` that does the following:

1. Stores a number to a variable.
2. Calculates the square root of the variable and stores it as a new variable.
3. Prints the results to the terminal (e.g. 'The square root of 9 is 3').

!!! Submit your script on Canvas as Exercise 5. !!!

Questions (you don't need to submit your answers)

Q. When are quotation marks needed and when should they be excluded in the `print` function?

A. Quotation marks are needed around strings and typically should not be included around variables and numbers (integers and floating point).

Q. When are parentheses needed in the `print` function?

A. Always in Python3!

Q. What are the differences between single, double, and triple quotes?

A. Single and double quotes are essentially interchangeable. Triple quotes are used when printing something accross multiple lines.

Q. What does a `#` signify when written directly before code on the same line? If unsure, try modifying the `first_script.py` by adding a `#` at the beginning of a line of code.

A. `#` is used to comment out code.

Q. In the Python interpreter or notebook, the output of a mathematical operation is returned without using the `print` function. What happens to the output of mathematical operations within a script? If unsure, test it.

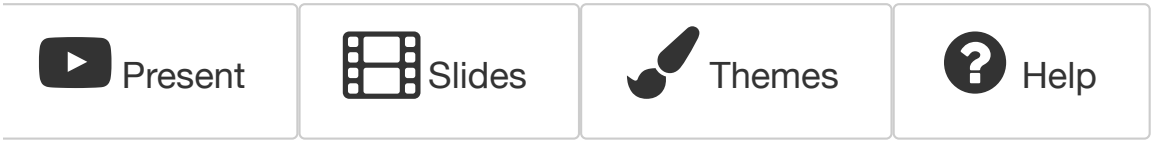
A. Values are returned into the void unless directed elsewhere.

Practice problems

Take the remainder of class to become more familiar with Python. Use Jupyter notebook, scripts, and the Python interpreter to store variables, do math operations, and print messagess to the terminal. Some sample problems are below.

1. PCR is often dones in 96 well plates. More high-throughput PCR assays can be done in plates containing 4 times as many reaction wells. In the **Python intepreter**, calculate how many PCRs can beone in a plate with 4 times 96 wells and save the value as a variable. Print the variable, along with a meassage to the terminal window while in the Python intepreter.
2. Write a **script** that assigns the values 96 and 4 to two separate variables and then uses them in an math equation to solve the product. Print the result to the terminal window.
3. If you didn't already do so, modify the script from above with comments describing what is being done at each step.
4. Modify the script above so that it prints the results to the terminal window with a friendly message spanning two lines.
5. Modify the script above so that it consists of a single `print` function.

In [ ]:


Present
Slides
Themes
Help