Introduction to Python

Hello! And welcome to Streetcode!! It's great to have you here! If you're reading this, you're in the Intro to Hack class, ready to learn Python! This is probably your first class, so we're going to start by showing you what Python is, and why we learn it!

What is Python?

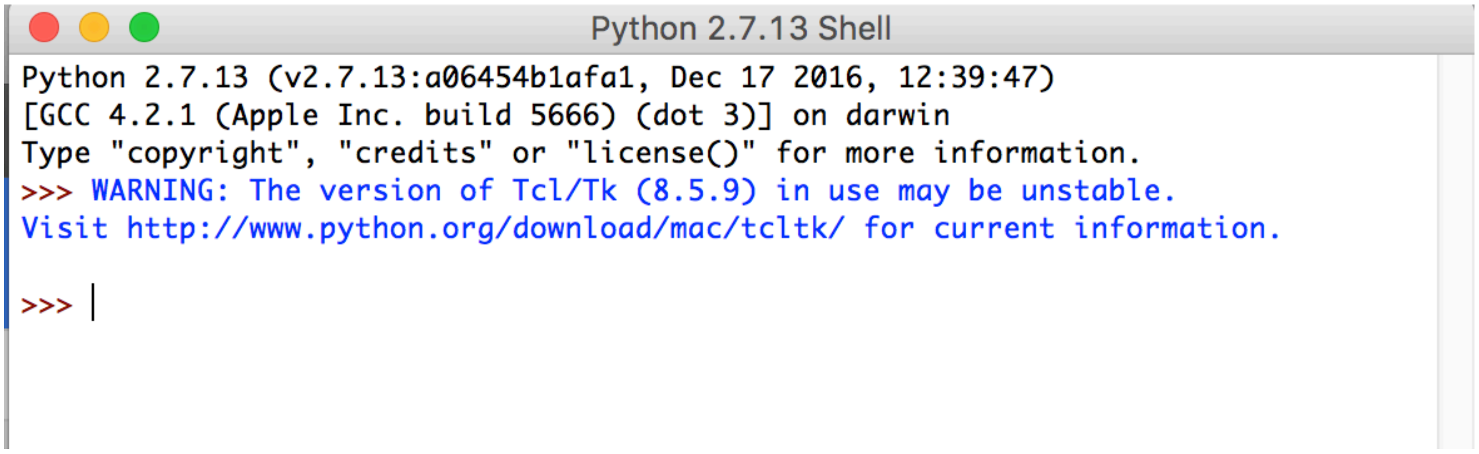
Have you ever wondered how people build apps, or create websites? Everything on your computer is powered by lines of instructions that tell the computer exactly what to do. We call these lines of instructions "code", and "coding" is nothing but the process of writing those instructions out one-by-one. Python is one of the "languages" used to give the computer instructions - and it's definitely one of the better ones! Here's what Python "code" looks like:

```
def draw(self):
        self.canvas.move(self.id, self.xspeed, 0)
        pos = self.canvas.coords(self.id)
        if pos[0] <= 0:
            self.xspeed = 0
        if pos[2] >= 500:
            self.xspeed = 0
```

If this looks a bit intimidating right now - don't worry! We're going to learn exactly how each of those lines work over the next 11 weeks. But if you really look at the code, you'll find it sort of looks a lot like English! The first line `def draw(self)` looks like its defining something (that's what the word "def" does!). The next line of code looks like it moves something, right? (`self.canvas.move`). The reason we learn Python is that it really does look a lot like English, and learning it is one of the best ways to learn how to program a computer!

First lesson - using IDLE!

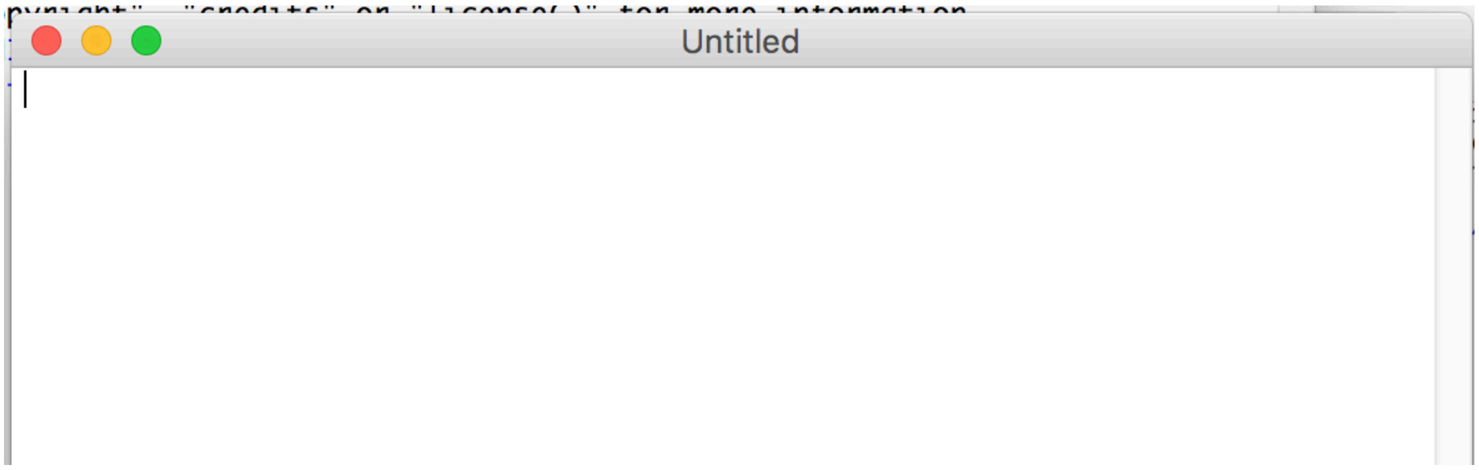
One of the hardest parts of programming is getting your computer to run your code! This might sound kind of silly - didn't we just say that all a computer does is run code? Well yeah, but getting it to run your code is actually kinda tricky. To run your Python code, we're going to use something called "IDLE". When you search your computer for IDLE and run it, you should see something like this:



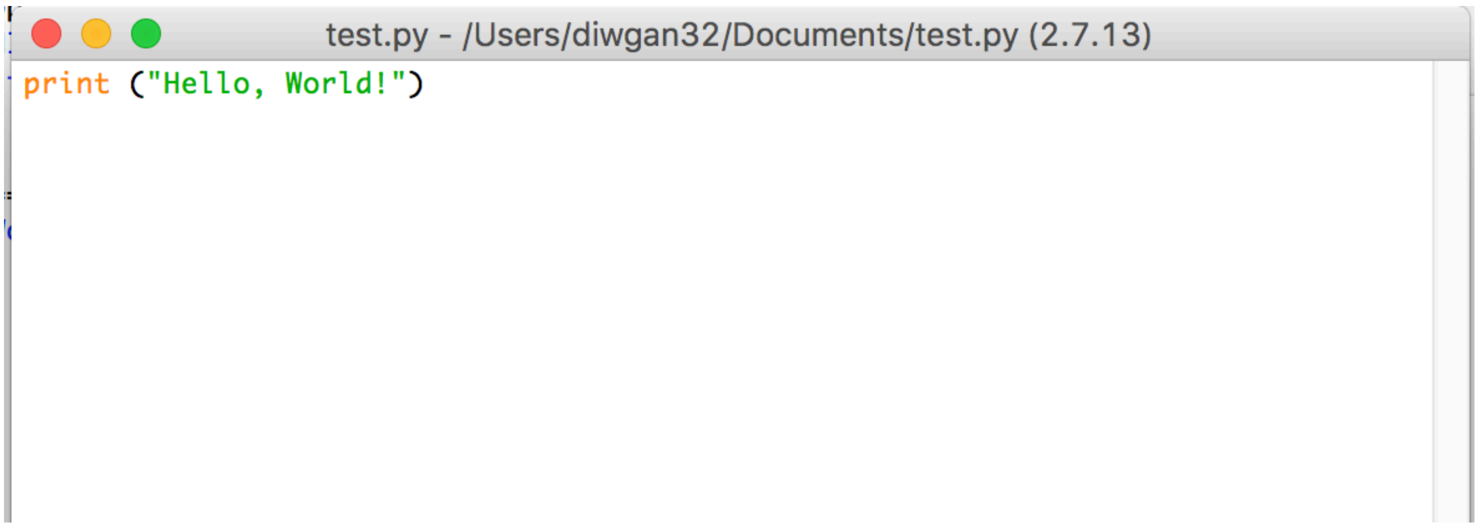
```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 12:39:47)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.

>>> 
```

IDLE is a program that "runs" your Python program. To get it to run a series of Python instructions (an instruction is just a line of code) that you wrote, click "File" and then "New File". This should open up a window like this:

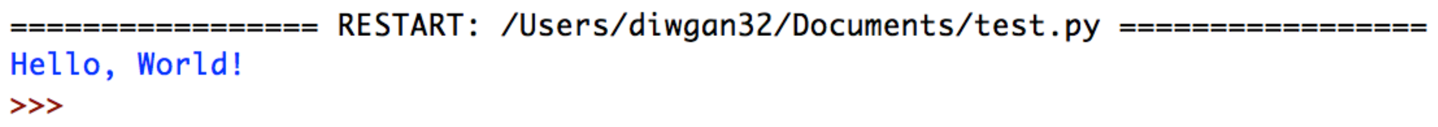


Now, we can get started writing code! For now, type the following line of code into this new window: `print ("Hello, World!")` , so that the window looks like this:



```
print ("Hello, World!")
```

Now, save this file onto your computer. Feel free to ask your teacher for help! I called my file "FirstPython.py". Then, click "Run" and then "Run Module". A new window should pop up that looks like this:



```
================= RESTART: /Users/diwgan32/Documents/test.py =================
Hello, World!
>>>
```

If it doesn't, ask one of the teachers for help. If it does, congrats :). You just wrote your first line of Python! Your on your way to creating whatever you want on a computer!

Syntax

Now, we have to talk about Syntax - or the way lines of Python code are put together. The first piece of syntax we're going to talk about is how to define **variables**. A variable is a way to store information - the different things you want to remember in your code. For example, if you wanted your code to "remember" your first and last name, you would write something like this:

```
FirstName = "Diwakar"
LastName = "Ganesan"
```

Notice the structure to each line. To create a **variable** in python, use the following pattern: variable*name* = value*. The equal sign in Python means something a little different than what you're used to in math. By saying* `my` `name = "Jack"`, I'm assigning "Jack" to the variable my_name. **Remember that the equal sign assigns variables in Python** Variable names have a few rules. First, you can't have spaces in a variable name. So, you *can't* do this:

```
First Name = "Diwakar"
```

But, you *can* do this:

```
First_Name = "Diwakar"
```

Notice the "_" character? It's called an underscore, and it can be used to replace spaces whenever you want spaces in your variable name.

Additionally, variable names **can't** start with a number. So, this wouldn't fly: `2Cool4Me = 5`.

Now, what can we store in variables? There are really only two different kinds of variables we can store in python: **numbers** and **text**.

Numbers

The most important thing you'd want your program to remember are numbers. To remember numbers in Python, you can use code like this:

```
my_age = 18
favorite_number = 32
temperature = 76.5
```

Notice how the names of the variables describe what kind of data it holds.

You can also write expressions with variables that involve numbers! For example, if I write:

```
my_favorite_number = 15
another_number = my_favorite_number * 2 - 10
```

Then, `another_number` will equal 15 (the value in `my_favorite_number` times 2 minus 10, so (15 * 2 - 10), which is 20. Again, we're assigning `another_number` to equal `my_favorite_number * 2`.

Text

Python also let's you remember short pieces of text, like your name! But, it's a bit tricky. You **can't** do this:

```
name = Diwakar Ganesan
```

Even though you followed the usual convention (variable_name = value), text needs to be surrounded by **quotations**. Text that's surrounded by quotations is called a **string**. So, the following line of code would work:

```
name = "Diwakar Ganesan"
```

You can also add together variables that contain text. For example, if I write:

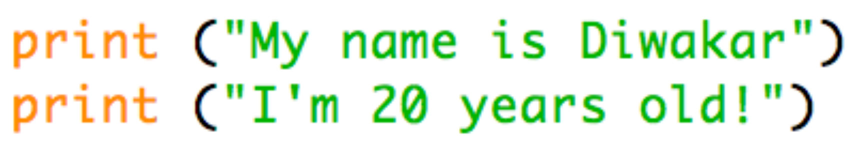
```
FirstName = "Diwakar"
LastName =  "Ganesan"
Name = FirstName + " " + LastName
```

Then, `Name` will contain the string `"Diwakar Ganesan"`. The last line of code used the **+** symbol to combine together three strings: `FirstName`, `" "` (a space character), and `LastName`.

Always remember: **text in python must be surrounded by quotes**.

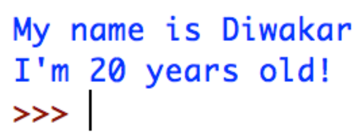
Print Statements

We're now going to talk about `print` statements! Remember the first line of code you wrote in this exercize? `print ("Hello, World')` . This line of code is called a print statement. A print statement is used to output lines of **text** to your screen! For example, if I write the following lines of code:



```
print ("My name is Diwakar")
print ("I'm 20 years old!")
```

Into a new IDLE script, and run the script, I should see the following output!



```
My name is Diwakar
I'm 20 years old!
>>> 
```

If you don't remember how to write a new Python script in IDLE, scroll up to the "First Lesson - using IDLE!" section or ask one of your teachers for help!

Each print statement consists of **4** parts:

- The word print: **print**
- An open parenthesis: **(**
- **Some Text**
- Close parenthesis: **)**

The trickiest part is figuring out what goes inside the open and close parenthesis. In the list above, it says to place **some text** in between the parenthesis. What this means is that you can print out anything that counts as a string. Above, in the variables section, we said that strings are any lines of text contained in quotation marks. So, all of the following are valid **print** statements:

- `print("Hello, World")`
- `print("I like " + "pie")`

  ```
  FirstName = "Diwakar"
  LastName =  "Ganesan"
  print (FirstName + " " + LastName)
  ```

  The last example may seem a little weird, but remember since `FirstName` and `LastName` are strings, you are allowed to print out the larger string `FirstName + " " + LastName` .

Practice Activity

Now that we've learned a bit about Python and it's syntax, let's practice! Ask your teacher for a Syntax Puzzle Kit! You'll receive a set of index cards that have scrambled lines of Python. The set of cards you recieve are shuffled, and you should use what you've learned so far to piece together **5** lines of Python code!