Python Basics + A few new concepts

Corey Fuller

Welcome to ACM E-Python Workshop!

- Even though we may not meet physically for the rest of the semester the Sessions will still continue!
- How this will happen
  a. The slides will be posted on Github along with an interactive workout(tentatively)
  b. The workbook will have some sort of program at the end that will produce output, which you will paste into the attendance form for the extra credit
  c. There may or may not be a live kahoot depending on participation

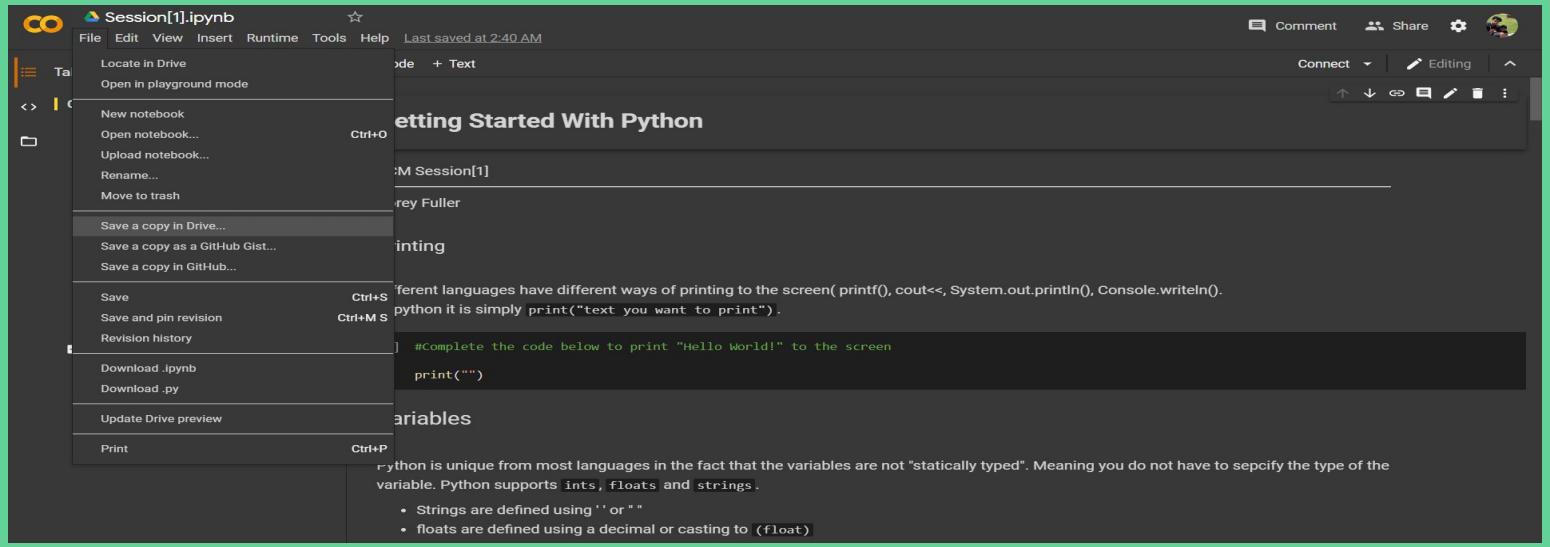
To Make Things Simpler...

- Go this link
  - Click the Session2 Folder
    - Click SESSION2.md
      - Click "Click here to access Session[2]'s interactive lesson file."
- All future slides will be posted on GitHub, uploaded to the discord, and sent via email.

https://github.com/coreyFuller/ACM-Python.git

Working with Google Colaboratory

- Google Colab allows us to make these sessions more interactive
- Once the notebook is open (you've clicked the link), save a copy of the notebook to your Drive
  - This will allow you to edit your own personal copy and follow along



Let’s Get Started!!!

Review

- This session will be relatively short compared to the ones we've had before
- This might be the standard as far as how E-Workshops will continue
- Try to fill out the review section of the google collab as much as you can
- Feel free to reference the previous session slides, Google, or use whatever resources necessary

New Material

Dictionaries

- Last Session we looked at a few ways data was stored in Python (variables, list...)
- **Dictionaries** are another way to store data in python
- Dictionaries work like maps in other languages and are built to support key/value pairs, removal, insertion, and searching
- Dictionaries are kind of like lists and arrays except the elements are stored in pairs
- Elements are stored with a Key(K) and a Value(V). The Key is used to as the index of the Value in the Dictionary
- A dictionary is statically declared like: dict_name = { K : V, K : V, K : V}

Dictionaries cont.

- Dictionaries do not have to be declared in one instance
- You can give your dictionary a name and add elements to it dynamically

```
Directory = {}

Directory["Corey"] = “cfulle3”

Directory["Jack"] = “jgwentworth”

Directory["Joe"] = “mama”

print(Directory)
```

Dictionaries

- The Key and Values in a dictionary can be **Any data type** and do not have to be the same data type as each other
  - Ex. String : int, int : list
- Iterating over a dictionary is done using ranged-based for loops
- Deleting elements in a dictionary is as simple as using the del keyword or the .pop() function like with lists
- Checkout the worksheet for examples and practice with dictionaries

Functions

- Python allows for the use of functions just like many other functions
- Functions are declared using the “def” keyword
  - Def my_funct()
- Instead of curly braces, we use “:” after the function definition and indentation to indicate what code belongs to the function
- Functions in python do not have a return type in the signature or the data type of the parameters
- To return data from the function you just use the “return” keyword

```
Def add(a, b):
    return a + b
```

Main

- Defining a main function in python is a little strange
- Check out the worksheet for an example of how to define main and practice with functions.

Classes and Objects

- Without prior object oriented programming experience, objects and classes can be a little tricky to grasp
- Hopefully most of you have had some experience with an object-oriented language (C++, Java, C#, Python)
- Classes serve as a blueprint for objects, and objects are made through grouping of related data and functions in a single entity known as encapsulation

Classes

- Classes are declared using the “class” keyword

```
class Triangle:
        Base = 12
        Height = 13
        def findArea(self)
```

- All class members in python default to public
- The “self” keyword is used so we can reference the data in the object itself

Objects

- Objects are just instances of a class
- They have their own copies of the data inside of the class
- Multiple objects can be created from the same class
- Objects are declared like:

```
myObject = myClass()
```

- The parentheses next to class name are just constructor calls
- If you've done OOP before great, if you haven't that's fine
- Hopefully the worksheet will provide enough practice

Paste the output from the worksheet into the last question to receive extra credit