Introduction to Python

GENERAL ASSEMBLY

Agenda

Activity	Topic
1	Welcome + Introduction
	Calendar & Expectations
	Intro to Slack
	Google Drive & Colab
	Github & Github Enterprise
2	Notebook 1: Intro to Python
3	Notebook 2: Generating an HTML file from Python
	Deploying a Static Web Page
4	Conclusion & Exit Tickets

Introduction

Python Programming

Python is the No. 1 fastest-growing major programming language with 151% year-over-year growth.

The future is bright for programmers who know Python — it’s a baseline skill for competitive industries like analytics, artificial intelligence, cybersecurity, and data science. And, thanks to its intuitive, readable syntax, it’s also one of the easiest languages to learn on the market.

When You Finish:

- Fluency in Python for Web Development and Data Analytics
- Knowledge of a wide range of tools used by Python developers
- Professional Portfolio of 10 Homework Projects
- Capstone Project

What we expect from you

Attendance:
17 out of 20 classes

Homework:
8 out of 10
assignments

Capstone Project

- Be present.
- Contribute constructively.
- Work hard.
- Ask questions.
- Be supportive.
- Talk to us!

Class Schedule

Date	Lesson	Topic
11/8/2021	1	Introduction to Python
11/10/2021	2	Data Structures
11/15/2021	3	Conditionals
11/17/2021	4	Loops
11/22/2021	5	Functions
11/24/2021	*No Class*	*No Class*
11/29/2021	6	Modules and Scripting
12/1/2021	7	Object-Oriented Programming
12/6/2021	8	Error Handling and Debugging
12/8/2021	9	Fundamentals Review Lab
12/13/2021	10	Fundamentals Flex Session
12/15/2021	11	Exploratory Data Analysis

Date	Lesson	Topic
12/20/2021	12	Data Visualization
12/22/2021	13	Cleaning and Combining Data
12/27/2021	*No Class*	*No Class*
12/29/2021	*No Class*	*No Class*
1/3/2022	14	Data Analysis Lab
1/5/2022	15	Data Analysis Review
1/10/2022	16	APIs
1/12/2022	17	Server Development with Flask
1/17/2022	*No Class*	*No Class*
1/19/2022	18	Flask Templates
1/24/2022	19	Web Development Lab
1/26/2022	20	Capstone Project Presentations

Tools for the Course

GitHub

GitHub Enterprise

HEROKU

Google Drive

python™

jupyter

HTML

plotly | Dash

colab

slack

Intro to Slack

- Primary tool of communication
- Ask questions publicly!
- Post completed homework in Slack
- Review & respond to others
- Use DM’s sparingly

Intro to Google Drive & Google Colab

- Most course material located here
- We’ll use Colab to run our code

Intro to Github & Github Enterprise

- Overlaps a little with Google Drive
- Useful for storing & collaborating on code
- Deploying static web pages and python applications
- Nearly all programmers use github!
- Github Public vs. Github Enterprise

Notebook 1

Introduction to Python

Overview

In this lesson, students will be introduced to executing Python code in a Jupyter Notebook environment. They'll learn variables and data types, along with string formatting and concatenation.

Duration

120 minutes

Learning Objectives

In this lesson, students will:

- Explain the value of Python.
- Use Jupyter Notebook to execute basic Python programs.
- Use operators to define and manipulate variables.
- Differentiate between data types in Python.

Our Learning Goals

- Explain the value of Python.
- Use Jupyter Notebook to execute basic Python programs.
- Use operators to define and manipulate variables.
- Differentiate between data types in Python.
- Deploying a static web page

What We’ll Practice Today

This class is a **blended learning experience**. It connects to and reinforces topics that you encountered in the myGA pre-work.

We’re going to return to topics covered in the pre-work and build upon them:

- Creating and manipulating **variables**.
- Python **data types**.
- Generating an HTML file & deploying it as part of a static web page

Guided Walk-Through:

1.1 Our First Jupyter Cell

10 minutes

Throughout our Python journey, we'll be using an interactive Python environment called **Jupyter Notebook** to accompany our lessons with coding exercises.

Let's open the notebook associated with this lesson and execute the first cell to learn more about Python's founding principles, known as the Zen of Python.

Solo Exercise:

Jupyter Notebook Review

10 minutes

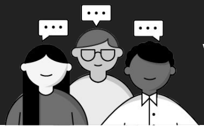
Let's dive right in and use what we learned in the pre-work! We want to understand where you are in your learning journey so that we can give the best possible experience in class.

Look over the exercises in today's Jupyter Notebook and attempt any that seem immediately doable to you.

Then, **rate your confidence level** on today's subjects from 1–5.

Discussion:

Why Python?



There are plenty of great programming languages out there.

Why are we learning Python?

Why Python?

- Python has grown as a high-level, general-purpose programming language with a **huge open-source community** supporting it.
- It’s the fastest-growing programming language on the market, especially within the data analysis community.
- Python's primary advantages:
    - **Clean syntax**.
    - **A wealth of specialized, pre-built libraries**, such as Pandas for data analysis and Django for web development.

Introduction to Python

Variables and Data Types

Variables

Variables are names that have been assigned to specific values or data.

Python allows us to easily define and redefine variables using a simple **assignment operator** (equals sign).

```
best_programming_language = "python"
```

Variable name

Value stored by the variable

Restrictions on Variables

- Variable names **cannot be just a number** (i.e., 2, 0.01, 10000).
- Variables **cannot be assigned the same name as a default or imported function** (i.e., “type,” “print,” “for”).
- Variable names **cannot contain spaces**.

Best Practices for Variables

- Variable names in Python should be **lowercase**.
- A variable's name should be **indicative of the concept it represents in your program**. Coming up with sensible variable names can take some time and thought, but this will save you a lot of confusion later on.
- If you have to include multiple words in your variable name, use an **underscore** to separate them. This is known as **snake case**.

Primitive Data Types in Python

	Explanation	Examples

String	A collection of characters representing a text-based value, such as a message.	`my_planet = "earth"`
		`secret_password = "password"`

Integer	Any whole number without decimal points.	`heist_members = 11`
		`bakers_dozen = 13`

Float	Numbers including points after the decimal, or "floating point" numbers.	`gigawatts = 1.21`
		`low_low_price = 49.99`

Boolean	The concept of true or false values.	`python_is_readable = true`
		`programming_is_simple = false`

Knowledge Check!

Match the values on left with their correct data type on the right.

"Hello world"	A. Float
6	B. Boolean
7.2	C. String
True	D. Integer

Solo Exercise:

1.2 Introduce Yourself

10 minutes

In Cell 1.2, we've already set up a message printed to the console. However, if we were to execute it right now, you'd see an error. To get this message to work, we have to define the following variables with string type values:

- `greeting`
- `name`
- `mood`

Don't worry about fully understanding what we've done to have the message show up — we'll be covering that throughout this lesson!

Introduction to Python

Manipulating Variables

Operators

We've already seen the **assignment operator** in action, but there are other operators that can modify variable values:

Symbol	Name	Explanation
+	Addition	Adds numbers or strings.
-	Subtraction	Subtracts numbers.
*	Multiplication	Multiplies numbers or strings.
/	Division	Divides numbers.
%	Modulus	Produces the remainder from division.
**	Exponent	Raises the first number to the second number's power.

Concatenating Strings

You may have noticed that some of the operators, especially the addition operator, work on strings as well as numbers.

Adding two or more strings together is called **concatenation**.

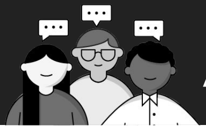
```
beverage_type = "sparkling water"

flavor = "grapefruit"

favorite_drink = flavor + beverage_type
```

Discussion:

A Problem With Concatenation



```
beverage_type = "sparkling water"

flavor = "grapefruit"
```

How could we concatenate these two variables into a single variable named “favorite_drink”?

There’s a good chance that our first solution will end up being "grapefruitsparkling water.” How can we fix that awkward combination of words?

Concatenation vs. String Interpolation

You can imagine that concatenating multiple strings might result in some complex, hard-to-read statements. However, Python provides a way to directly inject, or **interpolate**, a variable directly into a string using **f-strings**.

```
greeting = "Hello there"

person = "Professor Park"

message = f"{greeting}, {person}."
```

The message variable equates to “Hello there, Professor Park.”

Updating Variables

Variables wouldn’t be very useful if their values couldn’t vary or change. You can use the assignment operator to re-assign values to an existing variable.

```
favorite_language = "SQL"

favorite_language = "python"
```

You can even use the previous value when re-assigning!

```
my_current_age = my_current_age + 1

# Happy Birthday!
```

A Quick Comment on Comments

You’ve also seen that our Jupyter Notebook contains some lines of text that start with a hashtag (#). These are known as comments.

Comments are used to provide explanations and guidance throughout a program; Python will not try to execute these as code.

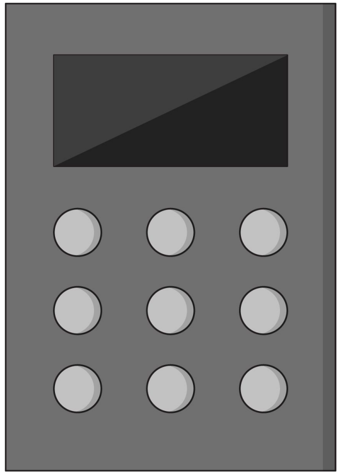
```
# This is a comment and will not cause errors!
```

Solo Exercise:

1.3 Operators and Updating Variables

10 minutes

Let's practice with a birthday calculator trick in the Jupyter Notebook by following the instructions in the comments.



All the Values That Are Fit to print()

We’ve seen a few examples using the print() function so far.

We’ll learn more about functions later on, but print() is an important first function to know. It allows us to output messages to the console, which can be extremely valuable for debugging a program.

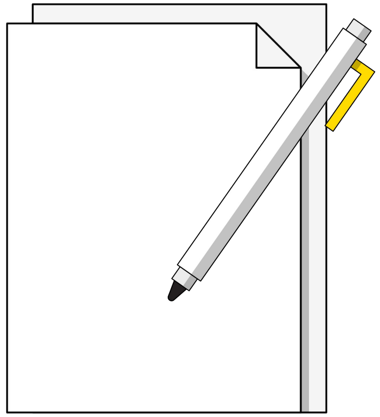
If something isn’t working, we can always use some print() statements to investigate whether or not our variables contain the values we expect.

Group Exercise:

1.4 Python Mad Libs

10 minutes

In this exercise, we'll practice combining strings together and including variables in string messages.



Introduction to Python

Changing Data Types

You’re Just Not My type()

One significant cause of errors in Python occurs when you’re trying to perform an operation on variables of different types:

```
2 + "apple"

# This will give you an unsupported operand error!
```

If you’re unsure what data type a variable contains, you can use the type() function to investigate, especially in combination with print().

```
print( type(mystery_variable) )
```

...But I Can Change!

We can overcome incompatible data types by **type casting**, or changing the data type of a variable:

```
str(2) + "apple"

# This turns 2 into a string
```

This technique has its limits, however. If we tried converting “apple” to an integer, we’d get another error, as that just doesn’t make sense.

Discussion:

Compatible Types

Of the four data types we've learned so far, **which do you think can be converted into each other and which cannot?**

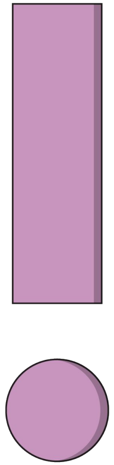
- Strings
- Integers
- Floats
- Booleans

Solo Exercise:

1.5 Code Fast and Break Things

15 minutes

Dealing with errors is part of everyday life in Python. Let's explore some common mistakes in Section 1.5 of the Jupyter Notebook.



Group Exercise:

1.6 How Many Ways to Print With Variables?

15 minutes

Researching how to do something new is an irreplaceable skill in programming. Even if you don't totally understand what's going on, you still need to be able to find code snippets in documentation and use them to solve problems!

Section 1.6 of the Jupyter Notebook challenges us to find four distinct methods of achieving the same objective — save us, Stack Overflow!

Notebook 2: Generating HTML & Deploying a web page

Introduction to Python

Wrapping Up

Recap

In today’s class, we...

- Explained the value of Python.
- Used Jupyter Notebook to execute basic Python programs.
- Used operators to define and manipulate variables.
- Differentiated between data types in Python.

Looking Ahead

On your own:

- Ensure that you’ve completed the Python pre-work and pre-work quiz.

Next Class:

Data Structures

Don’t Forget: Exit Tickets!