

# Introduction to Programming & Python

Brandon Krakowsky





1

---

---

---

---

---


---

---

---

## What is Programming?

- A *program* is a collection of instructions that performs a specific task (or set of tasks) when executed
- *Programming* is a way of specifying (or writing) the instructions
- *Programming languages* vary in many ways:
  - Syntax: Structure or grammar of the language
  - Semantics: Meaning of the code. What will it do when I run it?
  - Speed
  - Memory management
  - Etc.



Penn Engineering

Property of Penn Engineering | 2

2

---

---

---

---

---


---

---

---

## Client-Side vs. Server-Side Programming

- *Client-side programs* run on a *client*
  - Client-side programming has mostly to do with a user's interaction with a user interface
    - For example, a web page is a client-side program that runs in a web browser; the client
  - Common client-side programming languages are:
    - HTML, CSS, and JavaScript
- *Server-side programs* run on a *server* (or computer)
  - Server-side programming has mostly to do with the interaction between a user interface and a program on a server
    - For example, a web page sends messages (or requests) to a program on a server and it processes user input and interacts with a database
  - Common server-side programming languages are:
    - Python, Java, PHP, and ASP.NET



Penn Engineering

Property of Penn Engineering | 3

3

---

---

---

---

---

---

---

---

### What is Python?

- Python was named after the TV show *Monty Python's Flying Circus*
- Python is a *high-level programming language*
  - Provides abstraction from the details of the computer
    - Does most of the work in communicating with the computer
    - The code is intuitive and easy to understand
- Python is an *object-oriented programming (OOP)* language
  - Organized around objects rather than "actions"

Perin Engineering

Property of Perin Engineering | 4

4

---

---

---

---

---

---

---

---

### What is Python?

- Python is an *interpreted programming language*
  - Does not need to be *compiled*
    - Does not need to be converted from one language to another
    - For example: Java
  - Is *interpreted by a Python interpreter*
    - It's small and can run on any kind of computer!
  - This means that sometimes it's difficult to *debug* your Python programs
    - Do not make the mistake of typing out large chunks of code and not testing it at all

Perin Engineering

Property of Perin Engineering | 5

5

---

---

---

---

---

---

---

---

### Why Python?

- Python is an *open source* programming language
  - It's free!
- Python is *powerful, flexible, and intuitive*
  - There are many Python libraries and resources available online
  - Closely resembles the English language
- Python is *good for beginners* and a *great foundation* for other languages!
- Python can be used for:
  - Artificial intelligence/machine learning/natural language processing
  - Web development
  - Data analysis & visualization
  - Game programming
  - Desktop GUIs
  - *Many other purposes!*

Perin Engineering

Property of Perin Engineering | 6

6

---

---

---

---

---

---

---

---

## Configuring Python & Tools

Perrin Engineering

Property of Perrin Engineering | 7

7

---

---

---

---

---

---

---

## Download/Installing Python

- We will be using Python 3 in this course
  - If you already have Python 2 installed, please upgrade to Python 3
- To download and install Python, go here: <https://www.python.org/downloads/> (Download the latest version, do not download any version in "prerelease" status under maintenance)
- This download/install comes bundled with IDLE (Python's Integrated Development and Learning Environment)
  - Includes an interactive Python *interpreter* and *script editor*
  - We'll eventually be using IDLE to write and run Python *scripts*

Perrin Engineering

Property of Perrin Engineering | 8

8

---

---

---

---

---

---

---

## Jupyter Notebook

- For today, we'll use Jupyter Notebook to write and run Python code
- Jupyter Notebook runs in a browser on your computer
  - Includes interactive Python *interpreter* and *script editor*
  - To install, download Anaconda, a data science platform. This will install Python and Jupyter Notebook all at once: <https://www.anaconda.com/download> (Download the latest version)
  - To run, open Terminal on Mac or Command Prompt on Windows and run: `jupyter notebook`
  - Or launch from the Anaconda Navigator



For reference: <http://jupyter.org/install.html>

Perrin Engineering

Property of Perrin Engineering | 9

9

---

---

---

---

---

---

---

### Using Jupyter Notebook – Keyboard Shortcuts

- To execute code in a cell in a notebook  
Select the cell and press **CTRL + Enter**
- To execute code in a cell in a notebook, and select the next cell  
Select the cell and press **Shift + Enter**
- To insert a cell above  
Select the cell and press **a**
- To insert a cell below  
Select the cell and press **b**
- To delete a cell  
Select the cell and press **dd**
- To get help with Jupyter Notebook (Keyboard shortcuts)  
Anywhere outside of a cell, press **h**
- To get help with a Python function  
Put cursor inside parenthesis of function, and press **Shift + Tab**

Perin Engineering

Property of Perin Engineering | 10

10

---

---

---

---

---

---

---

---

### Using Jupyter Notebook – Exporting a Python Script

- It's normal to write, run, and maintain all of your code in a Jupyter Notebook file (.ipynb)
- That said, you CAN export a Python script (.py) from a notebook file
  - Go to "File" --> "Download As" --> "Python (.py)"

Perin Engineering

Property of Perin Engineering | 11

11

---

---

---

---

---

---

---

---

### Python Help – Other Tools

- Other Python Tools (IDEs)
  - PyCharm: Python IDE
    - <https://www.jetbrains.com/pycharm/download/>
    - We'll use this tool!
  - Eclipse with PyDev: Python IDE for Eclipse
  - Repl.it: Online editor and interpreter for Python (and other languages)
  - Text Editors: Emacs, Vi, Sublime, etc.

Perin Engineering

Property of Perin Engineering | 12

12

---

---

---

---

---

---

---

---

### Python Help – Language Resources

- Python Language Resources
  - Python Language Reference: <https://docs.python.org/3/reference/index.html>

Perrin Engineering

Property of Perrin Engineering | 13

13

---

---

---

---

---

---

---

---

### Python Language

Perrin Engineering

Property of Perrin Engineering | 14

14

---

---

---

---

---

---

---

---

### How Do I Write Python?

- Use the most basic Python *print* command to output to the console

```
print("Hello World!")
print('Hello World in single quotes')
```
- You can concatenate (link together) characters and strings using the *print* command

```
print('Today', 'is a good day!')
```
- Change what ends the print statement. (This is normally `\n`, i.e. new line)

```
print('Good morning,', end = ' ')
print('Brandon!')
```
- Specify the separator between arguments to print. (This is normally `' '`, i.e. space)

```
print('Good night', 'Brandon', sep = ' ')
```

*Note: Commands in light blue can be typed directly into a Jupyter Notebook file*

Perrin Engineering

Property of Perrin Engineering | 15

15

---

---

---

---

---

---

---

---

### Basic Data Types

- Every value has a *type* associated with it
- Integer (int): Positive or negative whole number with no decimal points  
8  
-1
- You can do math  
2 + 3  
5 - 6  
2 \* 3
- Remember the order of operations. You can use parentheses ()  
3 + 5 - 2 \* 6  
(3 + 5 - 2) \* 6
- Use the *type* command to test if an object is an *int*  
type(99)

Python Engineering

Property of Python Engineering | 16

16

---

---

---

---

---

---

---

---

### Basic Data Types

- Float (float): A positive or negative number that contains a decimal point  
1.3  
23.0  
-5.1  
2 \* 3.5  
7 / 2.0
- Test if an object is a *float*  
type(0.1)

Python Engineering

Property of Python Engineering | 17

17

---

---

---

---

---

---

---

---

### Basic Data Types - Arithmetic Operators

- Arithmetic operators  
+ addition  
- subtraction  
\* multiplication  
/ division  
// integer division, divides and returns the largest whole number, discarding the fractional result  
(ex. 3 // 2 = 1)  
\*\* exponent  
% modulus, divides and returns the remainder (ex. 7 % 5 = 2)

Python Engineering

Property of Python Engineering | 18

18

---

---

---

---

---

---

---

---

### Basic Data Types - Division

- Division: Divides  
`3 / 2`  
`3.1 / 2`
- Integer division: Divides and returns the largest whole number, discarding the fractional result  
`3 // 2`  
`3.1 // 2`
- Modulus: Divides and returns *remainder*  
`3 % 2`  
`4 % 2`  
`3.1 % 2`

Perin Engineering

Property of Perin Engineering | 19

19

---

---

---

---

---

---

---

---

### Basic Data Types

- Boolean (bool): True or False  
`1 == 2`  
`1 < 2`  
`1.2 >= 1.2`  
`'Car' == 'Car'`  
`'Car' == 'car'`  
`'1' == 1`  
`'1' != 1`
- Every object in Python has a boolean value. Test if an object is True or False  
`bool(False)`  
`bool(True)`  
`bool(7)`  
`bool(7 == 0)`

Perin Engineering

Property of Perin Engineering | 20

20

---

---

---

---

---

---

---

---

### Basic Data Types - Comparison Operators

- Comparison operators compare values and determine their relationship  
`==` equal  
`!=` not equal (can also use `<>`)  
`<` less than  
`>` greater than  
`<=` less than or equal to  
`>=` greater than or equal to

Perin Engineering

Property of Perin Engineering | 21

21

---

---

---

---

---

---

---

---

### Basic Data Types

- How do we know that 500002 is an even number?  
`500002 % 2 == 0`
- Is 500003 odd?  
`500003 % 2 >= 1`

Perin Engineering

Property of Perin Engineering | 22

22

---

---

---

---

---

---

---

---

### Basic Data Types

- String (str): Characters enclosed within single or double quotes  
`'Nice!'`  
`"Nice" == "Nice"`
- Concatenate (link together) characters and strings using a +  
`'Wow!' + ' Python is cool!'`
- Test if an object is a str  
`type('yes')`  
`type("103")`  
`type(103)`

Perin Engineering

Property of Perin Engineering | 23

23

---

---

---

---

---

---

---

---

### Basic Data Types

- Printing multiple strings  
`print('Name:', 'Brandon', 'Krakowsky')`
- Printing a concatenated string  
`print('Name: ' + 'Brandon' + ' Krakowsky')`
- Printing strings with special characters  
`print('Brandon\'s last name is Krakowsky')`  
- In Python strings, the backslash (\) is a special character, also called the "escape" character  
- Prefixing a special character (e.g. single quote) with a backslash (\) turns it into an ordinary character

Perin Engineering

Property of Perin Engineering | 24

24

---

---

---

---

---

---

---

---



### Basic Data Types - Casting

- Converting from one data type to another  
`12374/621`
- Did you get something like 19.92592...? What if you cast it to an *integer*?  
`int(12374/621)`  
- Be careful, it will round DOWN the value to the nearest *integer*!
- If you really want to round a *float* to the nearest *integer*, you can use Python's built-in *round* function  
`round(12374/621)`
- You can cast from a *string* to an *integer*  
`int('1')`

PenEngineering

Property of Pen Engineering | 25

25

---

---

---

---

---

---

---

---

### Basic Data Types

- Printing with numbers  
`print(4 / 2)`
- Printing with strings and numbers  
`print('4 % 2 = ', 4 % 2)`
- Printing with strings and numbers concatenated  
`print('4 % 2 = ' + 4 % 2)`  
- This will return an error!  
- You're trying to add a str to an int
- Try casting  
`print('4 % 2 = ' + str(4 % 2))`
- Printing with strings and booleans concatenated  
`print('Is 4 even? ' + str(4 % 2 == 0))`

PenEngineering

Property of Pen Engineering | 26

26

---

---

---

---

---

---

---

---