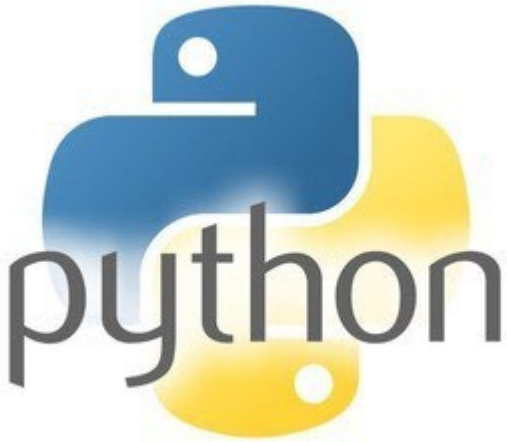
Python Bootcamp Week 1

Hello World



Vincent Y. Zhuang

Spring 2025

Why Bootcamp and why Python Bootcamp?

We believe that an ambitious student major in accounting and finance should learn at least one computer language.

The basic reason is that we have entered the Big Data era. In accounting and finance, we have a huge amount of data, and most of it is publically available free of charge.

To use such rich sources of data efficiently, we need a tool. Among many potential candidates, Python is one of the best choices.

Why Bootcamp and why Python Bootcamp?

Most of you do not have any programming experience before.

It’s absolutely okay if you are afraid of programming.

Programming is just a way of logic operations, from a machine perspective.

Understanding Programming from the Perspective of Machine

This course will make you in love with Python. Just like your lover…

Before we dive into detailed applications, we need to know how it works.

Why AF3214?

In a nutshell, we use real-world data for various financial topics.

Other programming courses normally use hypothetical data.

You've probably learnt numerous financial charts from previous courses.

We will teach you how to actually draw these beautiful charts using Python.

About this Bootcamp

Lectures plus tutorials (*optional*).

A live "lecture" and an *optional* in-class "tutorial". Answers will be discussed in the same lecture.

Class materials will be e-mailed to you before each lecture/tutorial (*optional*) from Week 1 onwards.

Python is a language

How do we talk to our computer in Python?

- **Interactive programming:** through a shell (interpreter, console) one line at a time
- **Batch programming:** running a whole script (a plain text file containing one to many lines of code)
- With the help of a GUI (graphical user interface). GUIs for coding are called **IDEs (Integrated Development Environments).**

Python IDEs - selected

IDLE
Python Software Foundation License

PyCharm
Proprietary software

Spyder
MIT License

Visual Studio
Proprietary software

Thonny
MIT License

Atom
MIT License

PyDev
Eclipse Public License

Wing IDE
Proprietary software

eric
GNU General Public License

Python Tools for Visual Studio
Apache License

Komodo Edit
GNU General Public License

PyScripter
MIT License

KDevelop
GNU General Public License

Geany
GNU General Public License

PIDA
GNU General Public License

Cloud9 IDE
Freeware

DrPython
GNU General Public License

Leo
MIT License

PythonAnywhere
Proprietary software

Install and Launch Python

1. Go to http://www.python.org/download
2. Depending on your computer, choose the appropriate package, for example, Python 3.11.1.

Generally speaking, the following are the three ways to launch Python:

- From Python IDLE (GUI)
- From the Python command line
- From your command-line window

Install and Launch Jupyter Notebook

1. Go to https://jupyter.org/install

2. Install JupyterLab with `pip`: `pip install jupyterlab`
   Once installed, launch JupyterLab with: `jupyter-lab`

3. Install the classic Jupyter Notebook with: `pip install notebook`
   To run the notebook: `jupyter notebook`

Now everything will be done using Jupyter Notebook

Python is:

50% Syntax

- Which words to use
- Punctuation
- Order
- Indentation
- Shortcuts

50% Logic

- Which tools to use in which order
- Creative
- Specific to one problem

Try not to worry if you don't understand everything today

Looking things up

You do not have to memorize everything you learn.

Coders of all levels rely on Google to help them recall the syntax for a particular task.

Google will point you to the official Python documentation or forums like Stack Overflow.

Objects and functions

The two main concepts of Python.

Object: a particular piece of data (like a noun)

Function: something you can do to/with an object (like a verb)

Objects and functions

You can use certain functions with certain classes of objects.

Some functions are shared between classes of objects, and some are not.

Bootcamp 1’s objects

- Integer (whole number) 3
- Float (number with digits following a decimal point) 3.0
- Boolean (True or False)
- String (text or characters) “3” or “three”
- *Assigning objects to variables*
- *Indexing to get only part of an object*

Bootcamp 1’s functions

- Basic operators: + - * /
- Functions to convert between data types
- String functions

How to open Bootcamp_1_Lecture.ipynb:

Jupyter Notebook on your own computer:

- Download the file and Open Jupyter Notebook. **Navigate to the folder where you just stored the file.** Open Bootcamp_1_Lecture.ipynb

Let's code!

Variable names

- No spaces
- Case matters – k is not K
- Must start with a letter
- Should be meaningful

Style conventions:

- camelCase
- separate_with_underscores

In Python indexing, we start counting with 0

Indexing strings

```
"H  e  l  l  o     W  o  r  l  d  !"
 0  1  2  3  4  5  6  7  8  9  10 11
```

"Hello World!"[0] is "H"

"Hello World!"[4] is "o"

Indexing strings

```
"H  e  l  l  o     W  o  r  l  d  !"
 0  1  2  3  4  5  6  7  8  9  10 11
```

`"Hello World!"[6:12]` is `"World!"`

To take a substring, you start with the position of the first character you want included, and end with the position 1 past the last character you want.

Indexing strings

```
"H  e  l  l  o     W  o  r  l  d  !"
 0  1  2  3  4  5  6  7  8  9  10 11 12
```

`"Hello World!"[6:12]` is `"World!"`

This is because behind the scenes Python is actually labeling the invisible spot between each letter in our string.

Python functions

Parentheses immediately follow a function name.

```
print("Hello World")
```

The information inside the parentheses is called an **argument**.

We **passed** the argument "Hello World" to the print function.

Two types of Python functions

1. Can stand on its own. Takes an object as an **argument.**

   These types of functions usually do something ***with*** the object.

```
my_name = "Xyz"
print(my_name)          Xyz
len(my_name)            3
```

Two types of Python functions

2. Follows an object. Some take arguments and some don't.

   These types of functions usually do something ***to*** the object.

   These are sometimes called **methods** in Python.

```
my_name = "Xyz"
my_name.replace("X", "x")        "xyz"
my_name.upper()                  "XYZ"
```

Python functions

When a function takes multiple arguments, they are separated by a comma:

```
my_name.replace("X", "x")        "xyz"
```

How to open Bootcamp_2_Lecture.ipynb:

Jupyter Notebook on your own computer:

- Download the file and Open Jupyter Notebook. Navigate to the folder where you just stored the file. Open Bootcamp_2_Lecture.ipynb

Let's code!

Writing functions

Function calls

You already know how to **call** a function.

```
len(my_string)
```

```
open(my_file, "r")
```

Function definition syntax

This is the syntax to **define** a new function:

```
def function_name(any arguments needed):
    do something or create a new object
```

Function definition syntax

It is also good practice to include a **comment** that says what your function does. Comments start with a #and are ignored by Python.

```
def function_name(any arguments needed):
        #my function does something
        do something or create a new object
```

Let's code!

Bootcamp 1's objects

- ✓ Integer (whole number) 3
- ✓ Float (number with digits following a decimal point) 3.0
- ✓ String (text or characters) "3" or "three"
- ✓ Boolean (True or False)
- ✓ *Assigning objects to variables*
- ✓ *Indexing to get only part of an object*

Bootcamp 1’s functions

- Basic operators: + - * /
- Functions to convert between data types
- String functions