PYTHON SEMINAR 2020

JENS HAHN

THEORETICAL BIOPHYSICS

TODAY

I. git recap

II. Environments

III. Object orientation

I. GIT RECAP

github.com/user/repo

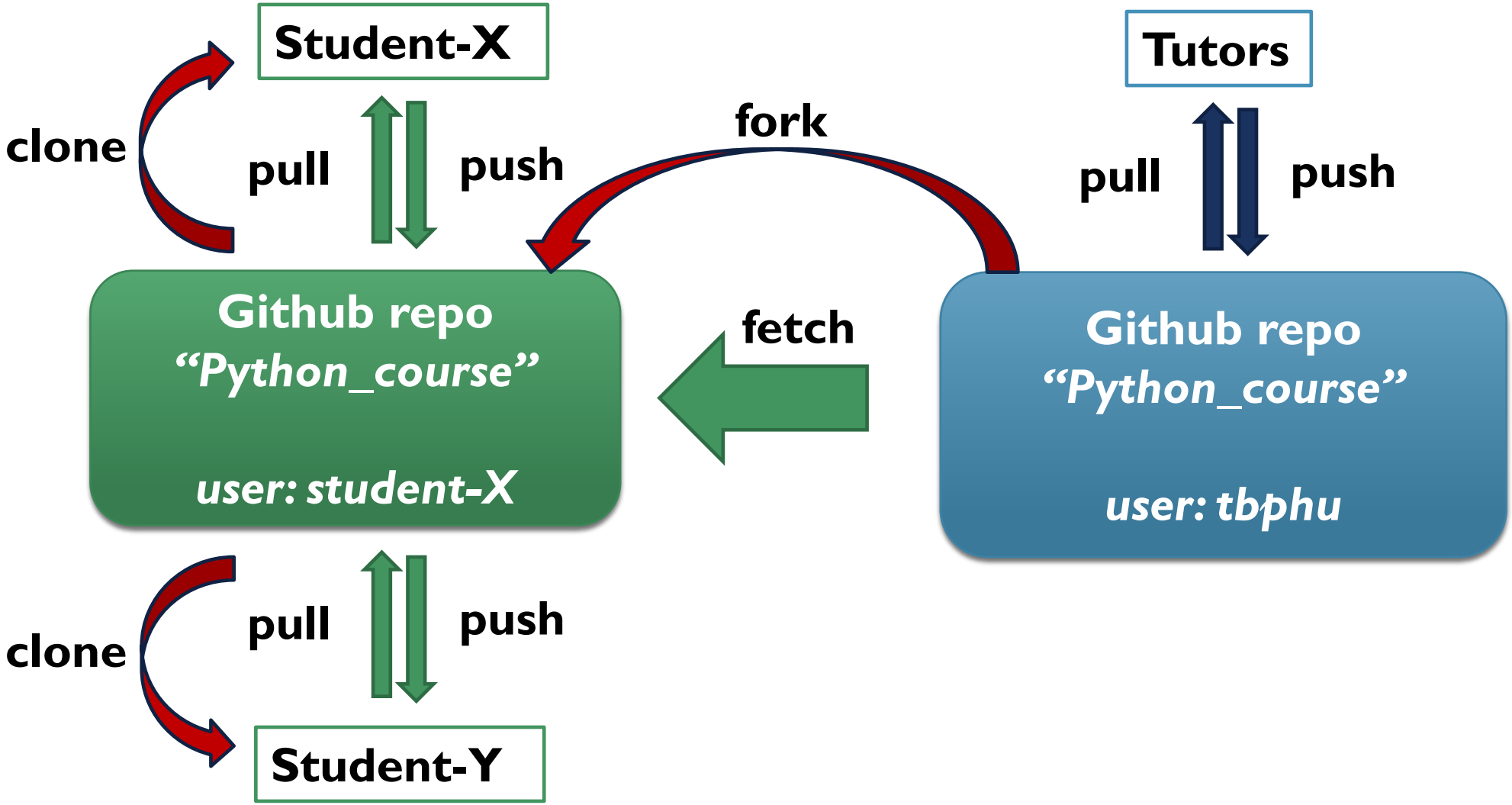
`remote repository`

`fork`

`remote repository`
github.com/me/my_repo

`clone`

`local repository`
/home/Documents/.../repo/


Student-X
clone
pull
push
Tutors
fork
pull
push
Github repo
"Python_course"
user: student-X
fetch
Github repo
"Python_course"
user: tbphu
clone
pull
push
Student-Y

I. GIT RECAP

Tasks

- Fork the repo online (on *github.com*) - get the course material
- Configure *git* (local) - tell *git* who you are
- Clone the forked repo (local) - download course material
- Set another remote repo (local) - update course material

I. LAST STEPS FOR GIT

- Set upstream : just a name

```
git remote add upstream https://github.com/tbphu/Python_course.git
```

git fetch upstream
git merge upstream/master

- Configure git: just once
  - `git config --global user.email "jens.hahn@hu-berlin.de"`
  - `git config –-global user.name "Jens Hahn"`

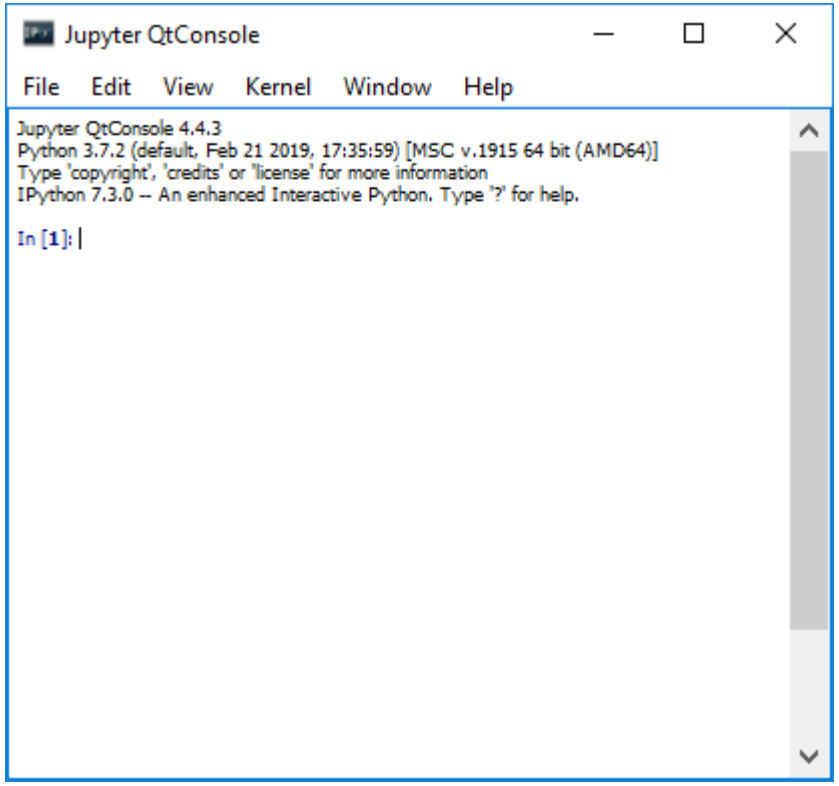
II. ENVIRONMENTS

Different ways to use Python:

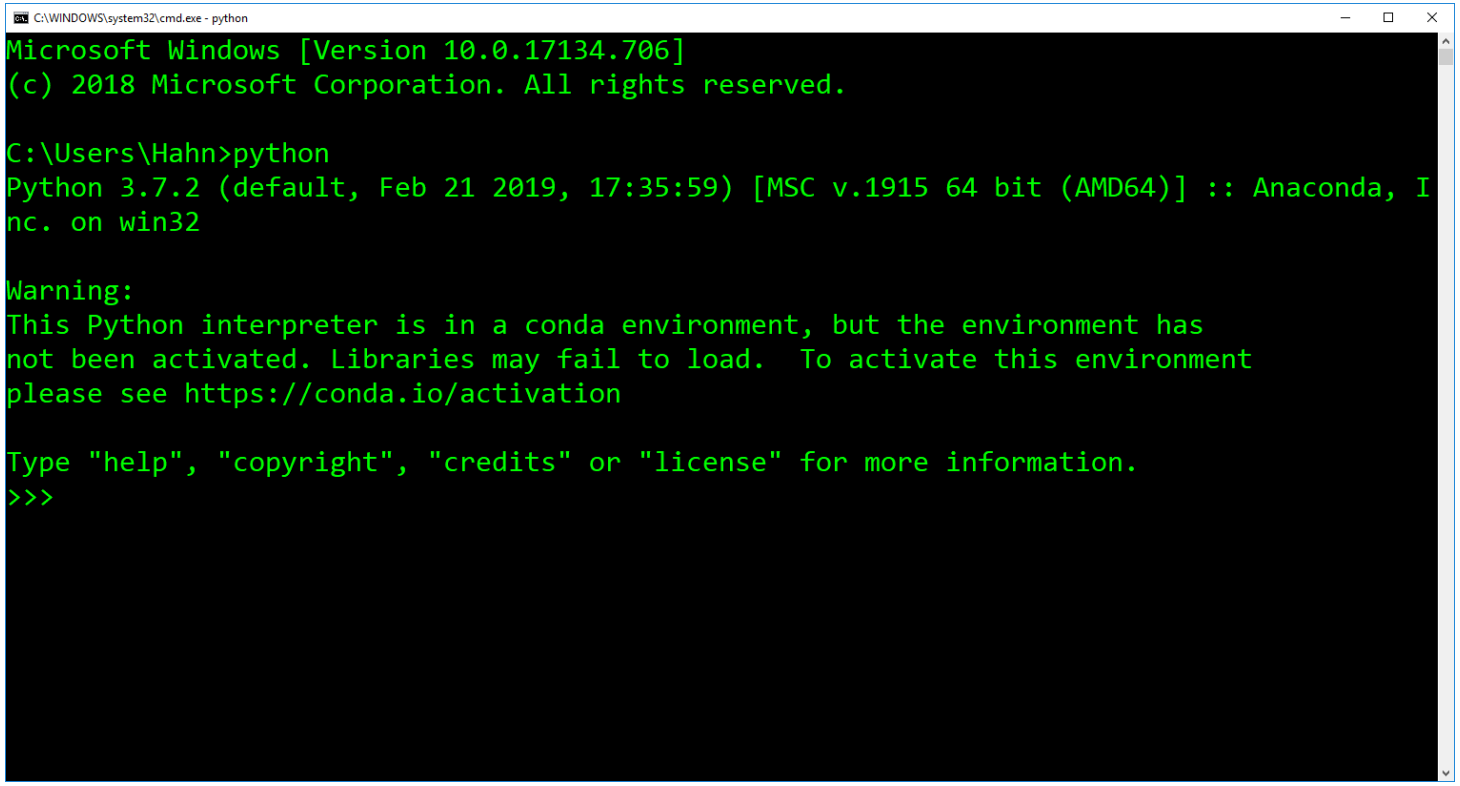
- Console - direct input, not saved - testing, fast
- Script - indirect input, saved - programs, complex
- Notebook - direct input, saved - documentation, I/O

II. CONSOLE

QtConsole



Command-line (Eingabeaufforderung)



II. SCRIPTING

Notepad, Notepad++

Untitled - Notepad

File Edit Format View Help

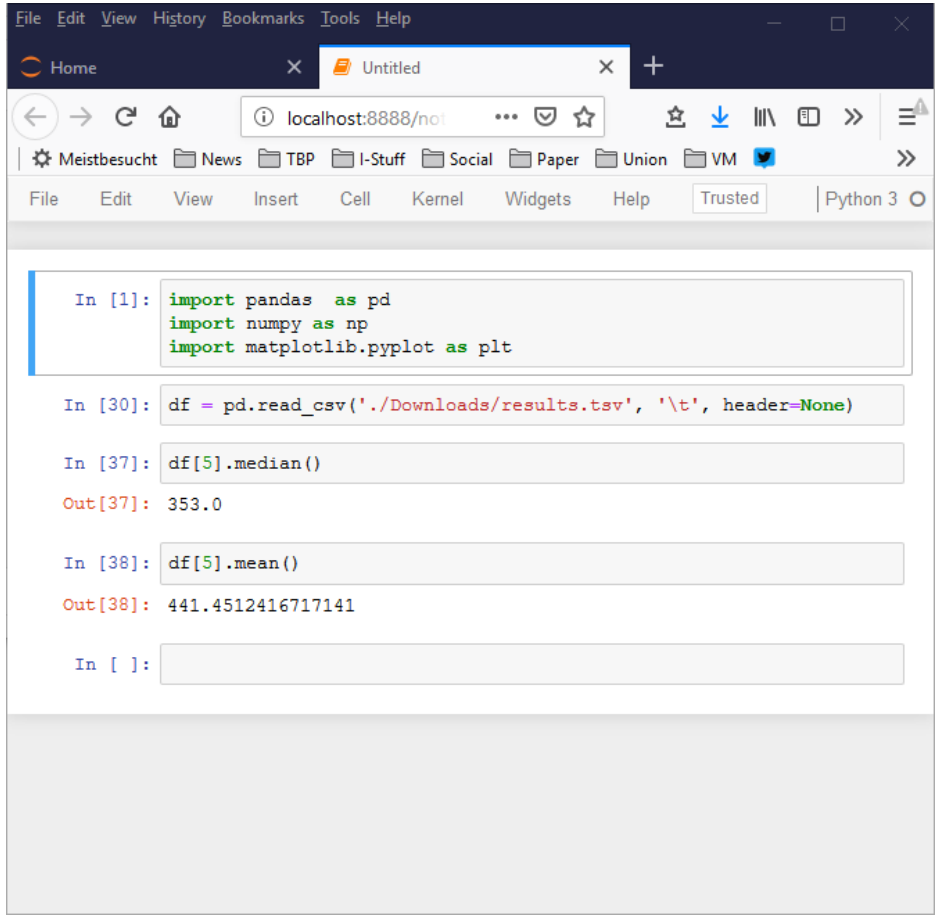
```
import numpy as np
import matplotlib.pyplot as plt

def meep():
    print('meep')
    return None
```

Sublime Text

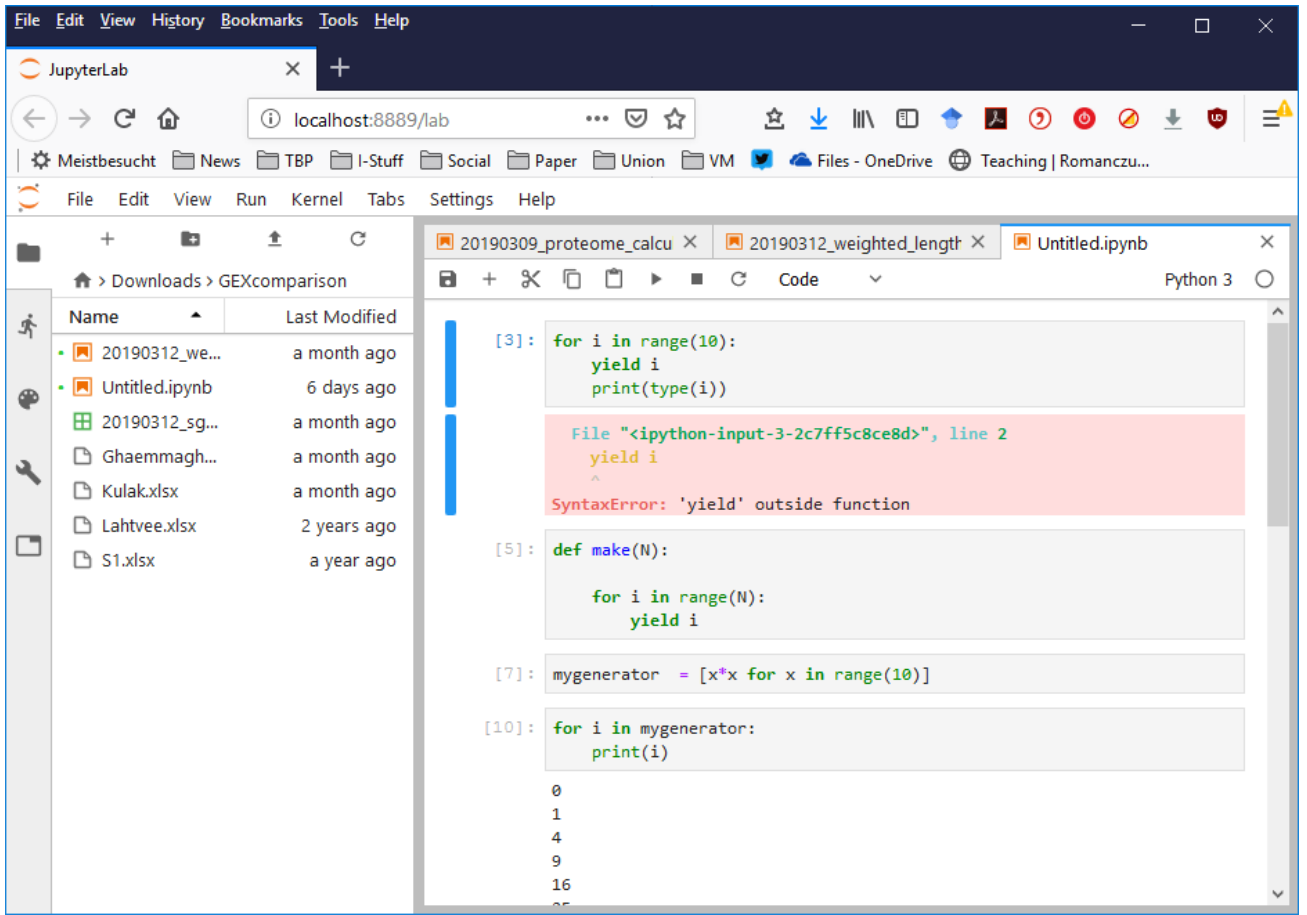
```
1  import matplotlib.pyplot as plt
2
3
4  measured_ccm = [1500, 1234, 1432, 1324, 1543]
5  simulated_ccm = [1600, 1114, 1632, 1111, 2333]
6
7  measured_cb = [200, 500, 333, 444]
8  simulated_cb = [188, 100, 123, 444]
9
10 fig = plt.figure()
11 plt.plot([0, 2500], [0, 2500], label=None, color='k', alpha=0.3, linewidth=3)
12 plt.scatter(measured_ccm, simulated_ccm, label='Central Carbon Metabolism')
13 plt.annotate('important enzyme',
14              xy=(1234, 1114),
15              xytext=(200, 1500),
16              arrowprops=dict(facecolor='black', width=2, headwidth=8))
17 plt.scatter(measured_cb, simulated_cb, label='Cofactor Biosynthesis')
18 plt.text(1800, 200, "I am a string.")
19 plt.xlim(0, 2500)
20 plt.ylim(0, 2500)
21 plt.legend()
22 plt.xlabel('Measured Protein Abundance [copies/cell]')
23 plt.ylabel('Simulated Protein Abundance [copies/cell]')
24 plt.show()
25
26
```

II. NOTEBOOK

Jupyter Notebooks

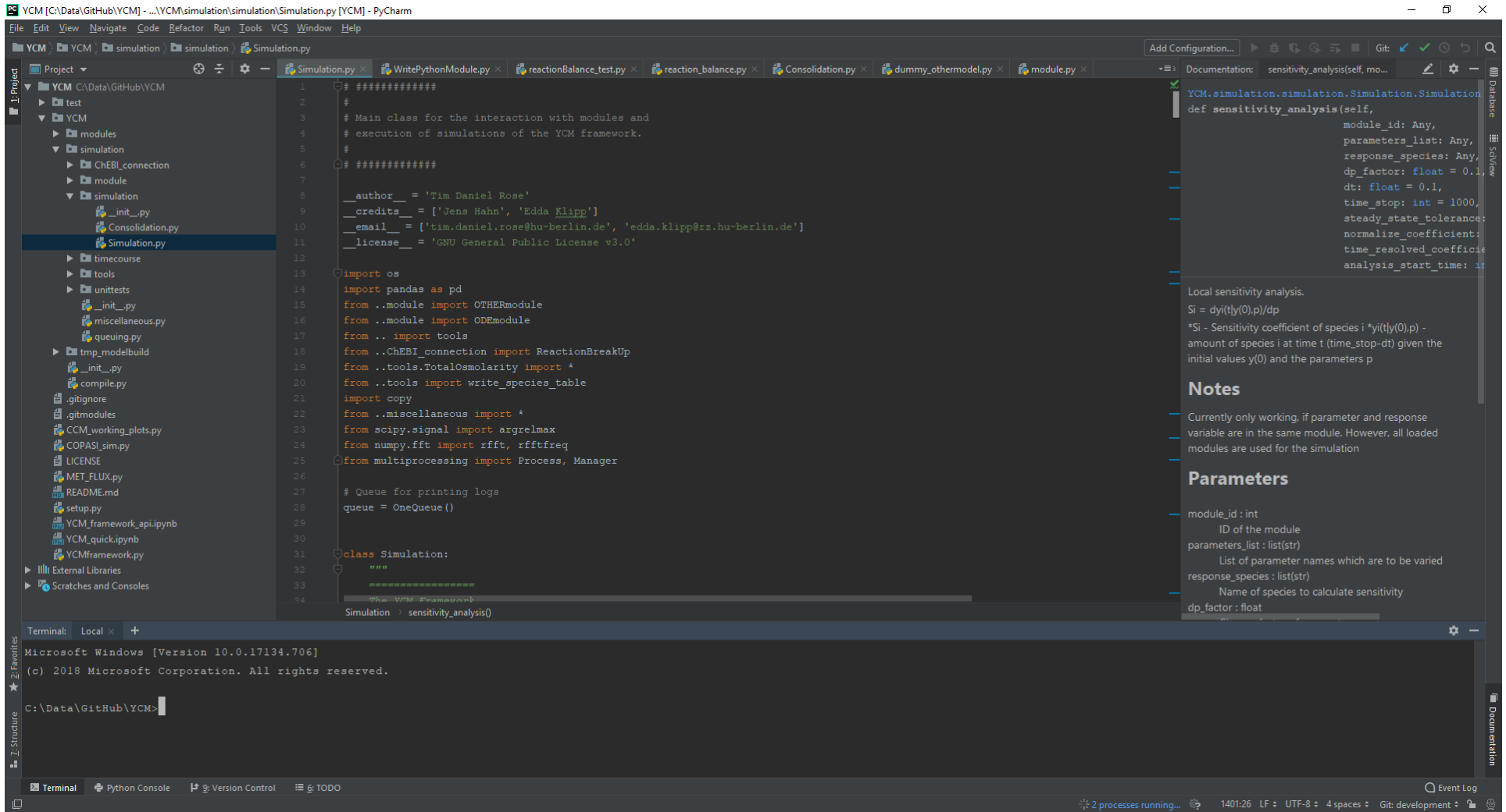


JupyterLab



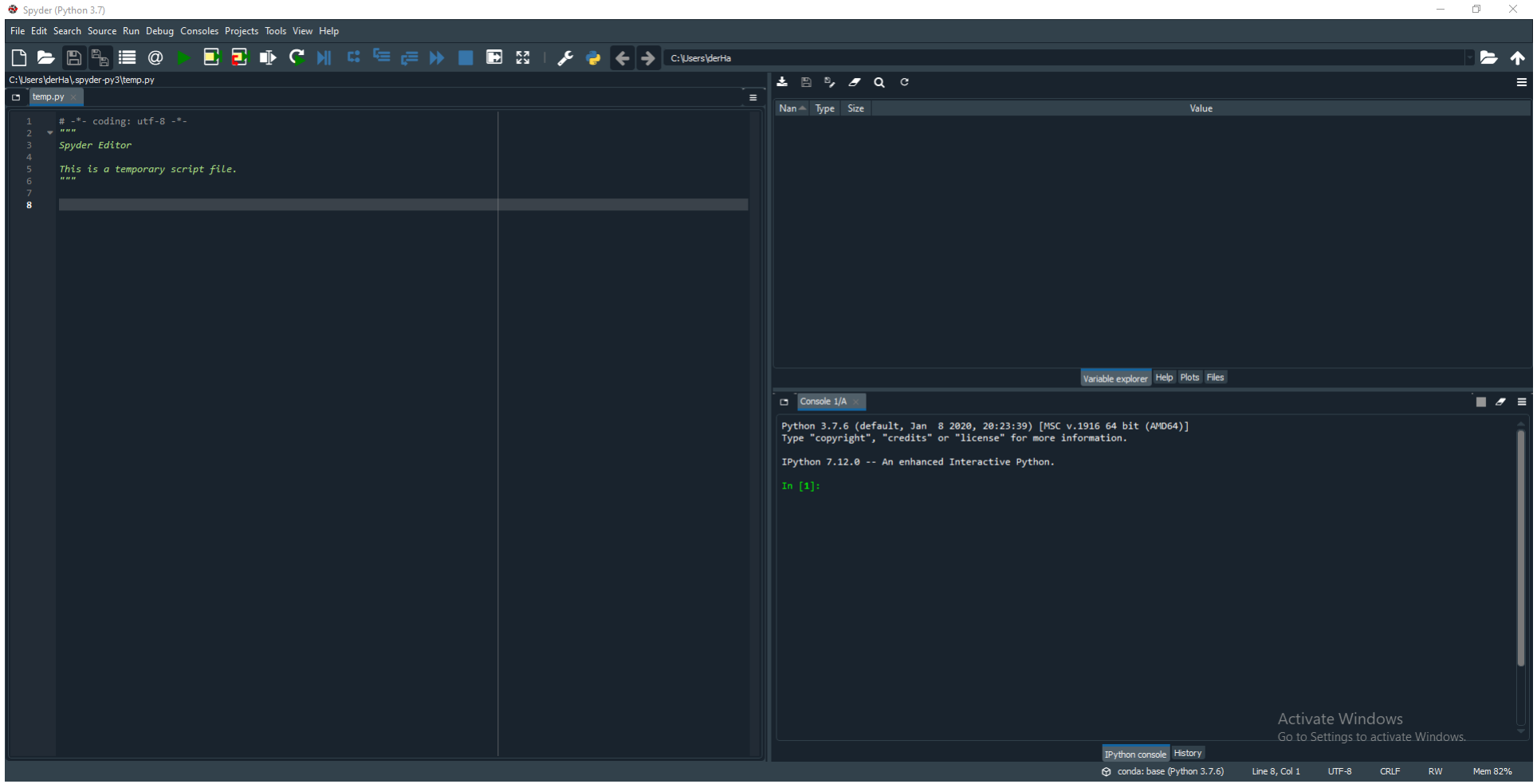
II. IDE Integrated Development Environment

PyCharm
(Jetbrains)



II. IDE Integrated Development Environment

Spyder (Anaconda)



III. HOW IT WORKS

Python code interpretation

- Python version 2 or 3?
- Python implementations (Cpython, PyPy…)
- How to get from source code to machine code?
- Interpreted or compiled language?

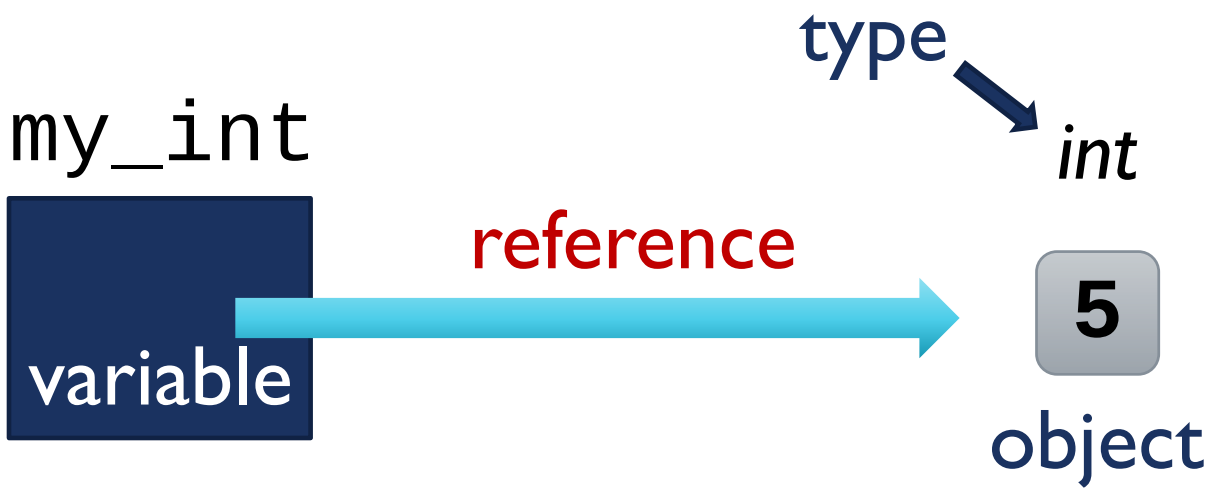
Python internals

- What is an *object* in Python?
- How are *object* types set?
- Mutable vs. immutable objects?

III. ASSIGNMENTS (ZUWEISUNGEN)

5 is more than just a number!!

```
my_int = 5
```

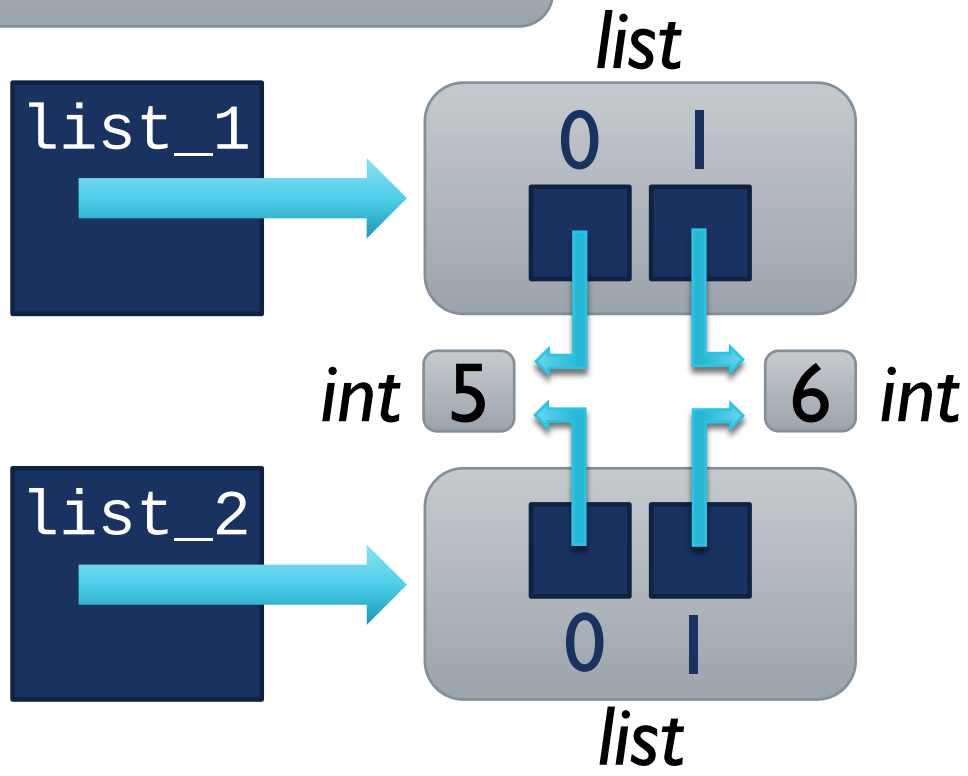


III. ASSIGNMENTS (ZUWEISUNGEN)

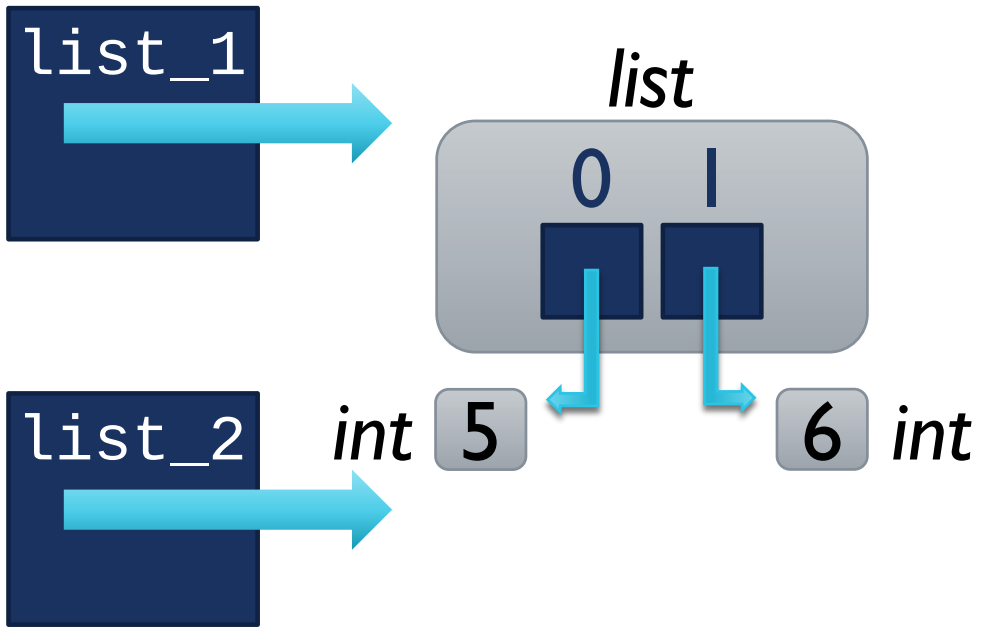
- Duck typing - Python choses the *object type* automatically!
- Object - Every *object* has *attributes* and *methods* (functions)
- Attention!! - *objects* can be **mutable** or **immutable**

III. ASSIGNMENTS (ZUWEISUNGEN)

```
list_1 = [5,6]
list_2 = [5,6]
```




```
list_1 = [5,6]
list_2 = list_1
```

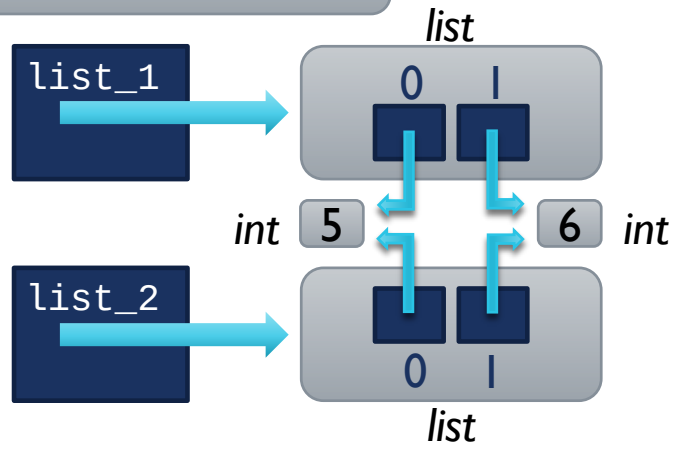


III. COMPARISONS (VERGLEICHE)

```
list_1 == list_2
```

```
list_1 is list_2
```

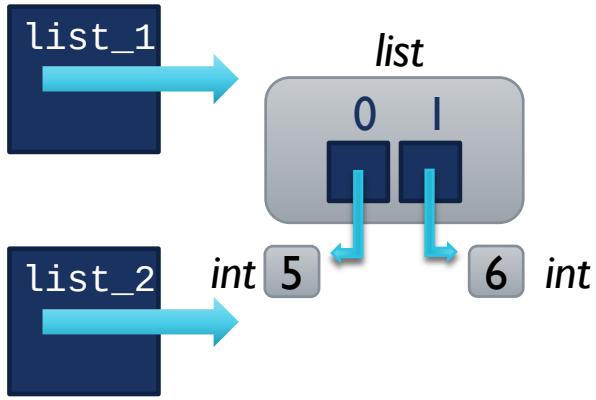
```
list_1 = [5,6]
list_2 = [5,6]
```




```
list_1 == list_2
```

```
list_1 is list_2
```

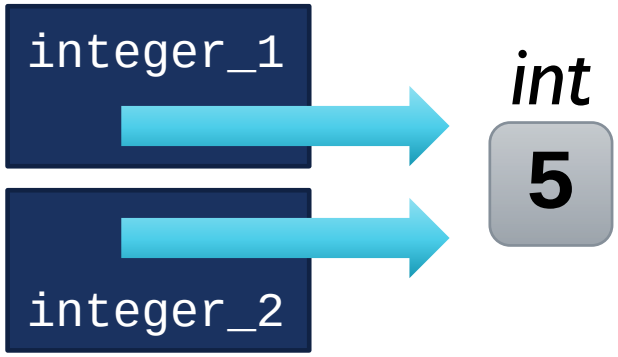
```
list_1 = [5,6]
list_2 = list_1
```



III. MUTABLE VS. IMMUTABLE

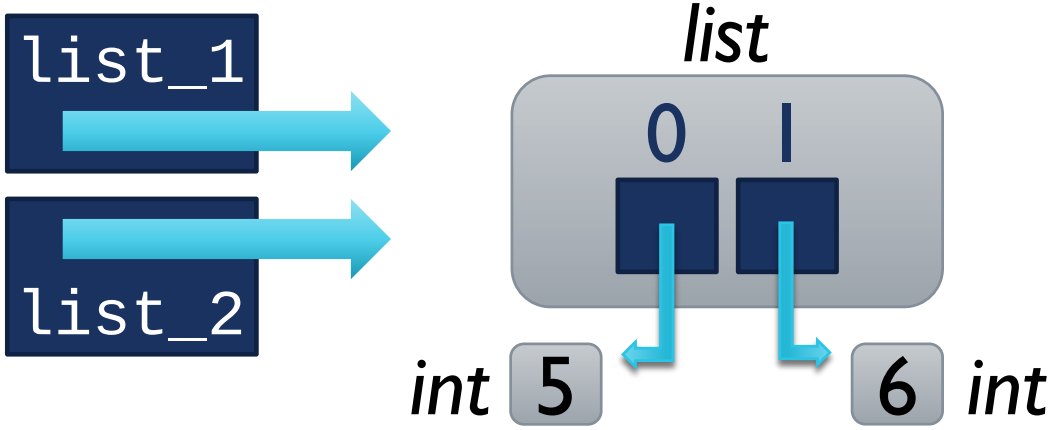
```
integer_1 = 5
integer_2 = integer_1
```

```
integer_2 = integer_2 + 1
```



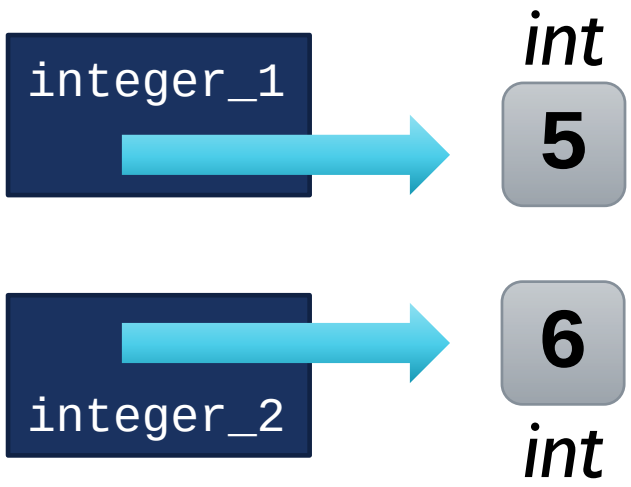

```
list_1 = [5,6]
list_2 = list_1
```

```
list_1.append(3)
```

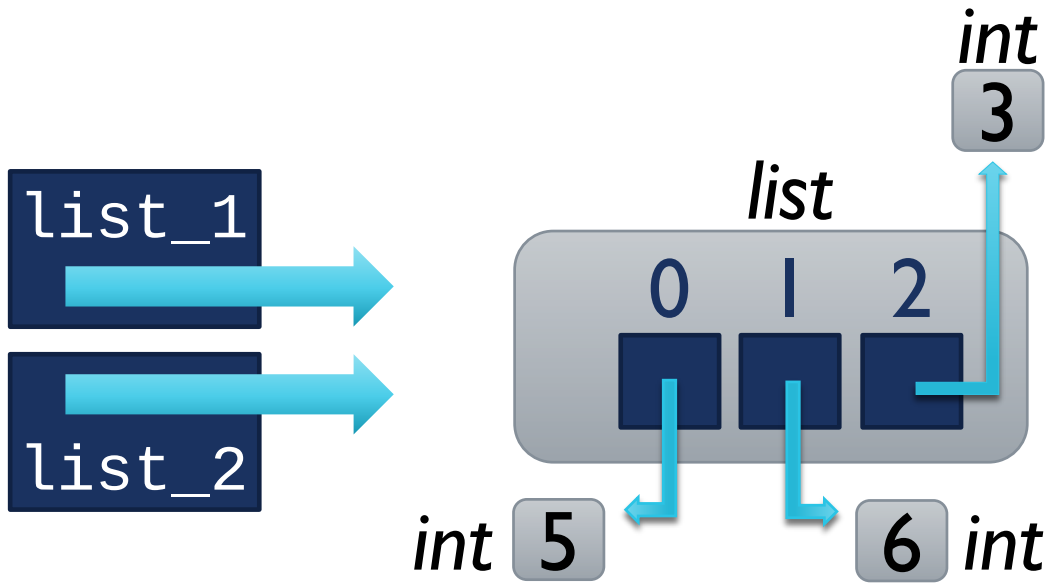


III. MUTABLE VS. IMMUTABLE

```
integer_2 = integer_2 + 1
```




```
list_1.append(3)
```



III. HANDS ON ASSIGNMENTS

- Fetch from *upstream* - get the assignment
- Open `assignment_1.py` - do the assignment
- Run file in Python - check your results
- Push to forked repo - get the assignment in YOUR repo

III. FURTHER READING

Python versions

- Pick a Python Interpreter

  https://docs.python-guide.org/starting/which-python/

- Interpreted or Compiled?

  http://net-informations.com/python/iq/interpreted.htm

Python internal

- The *id of an object in Python*

  https://www.programiz.com/python-programming/methods/built-in/id

- Mutable vs. immutable objects

  https://towardsdatascience.com/https-towardsdatascience-com-python-basics-mutable-vs-immutable-objects-829a0cb1530a