

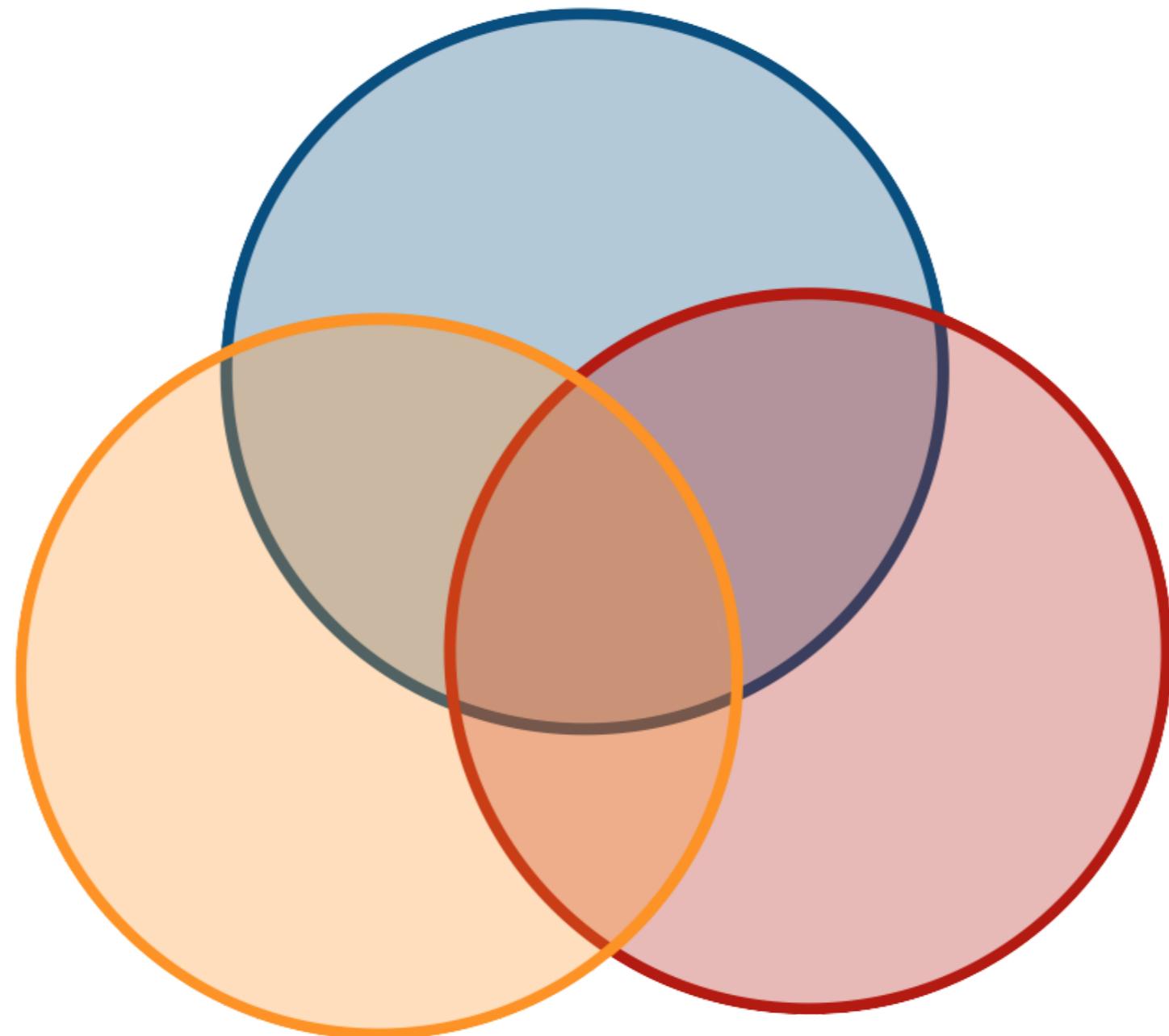
# Python, data science, & software engineering

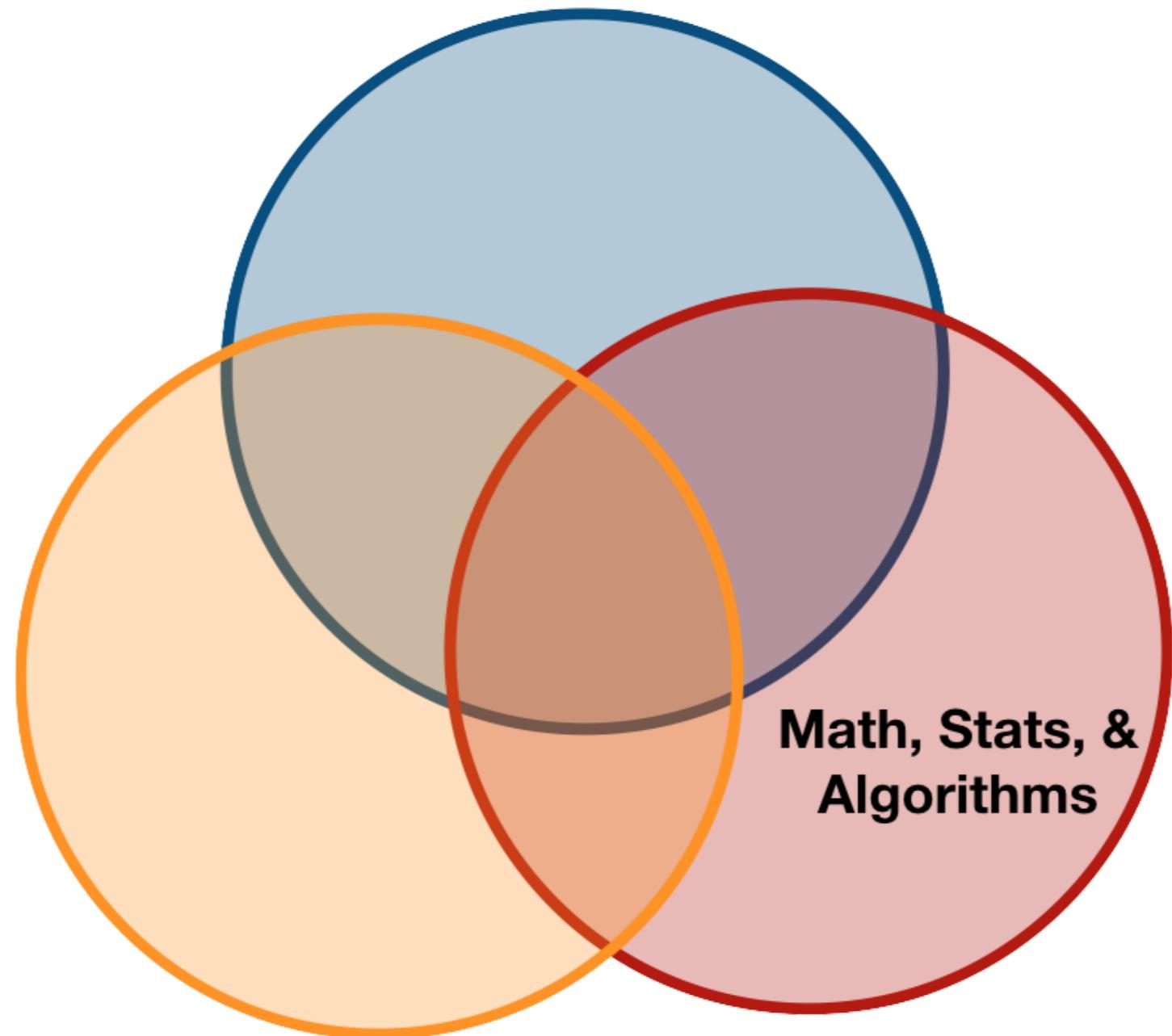
SOFTWARE ENGINEERING PRINCIPLES IN PYTHON

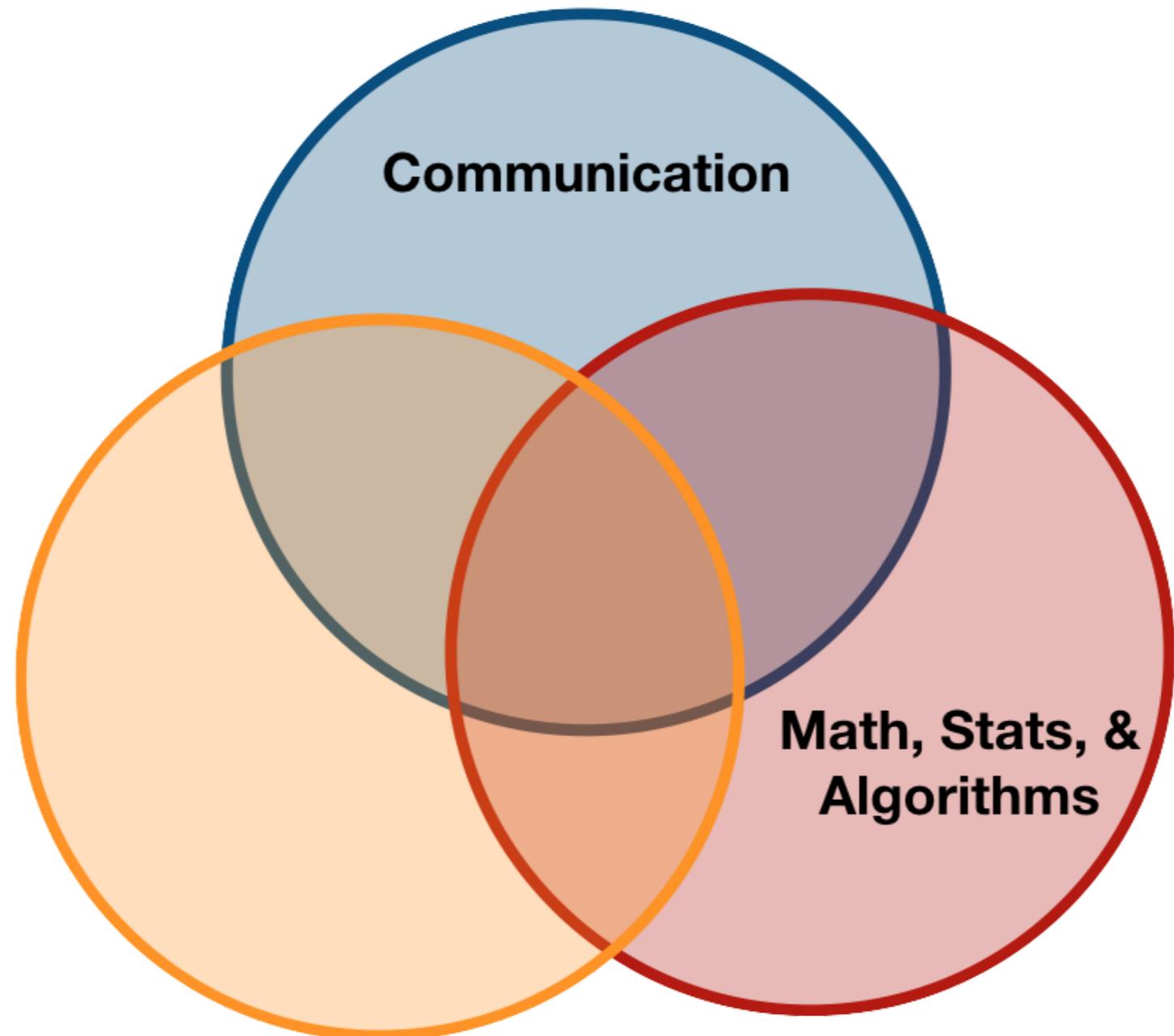


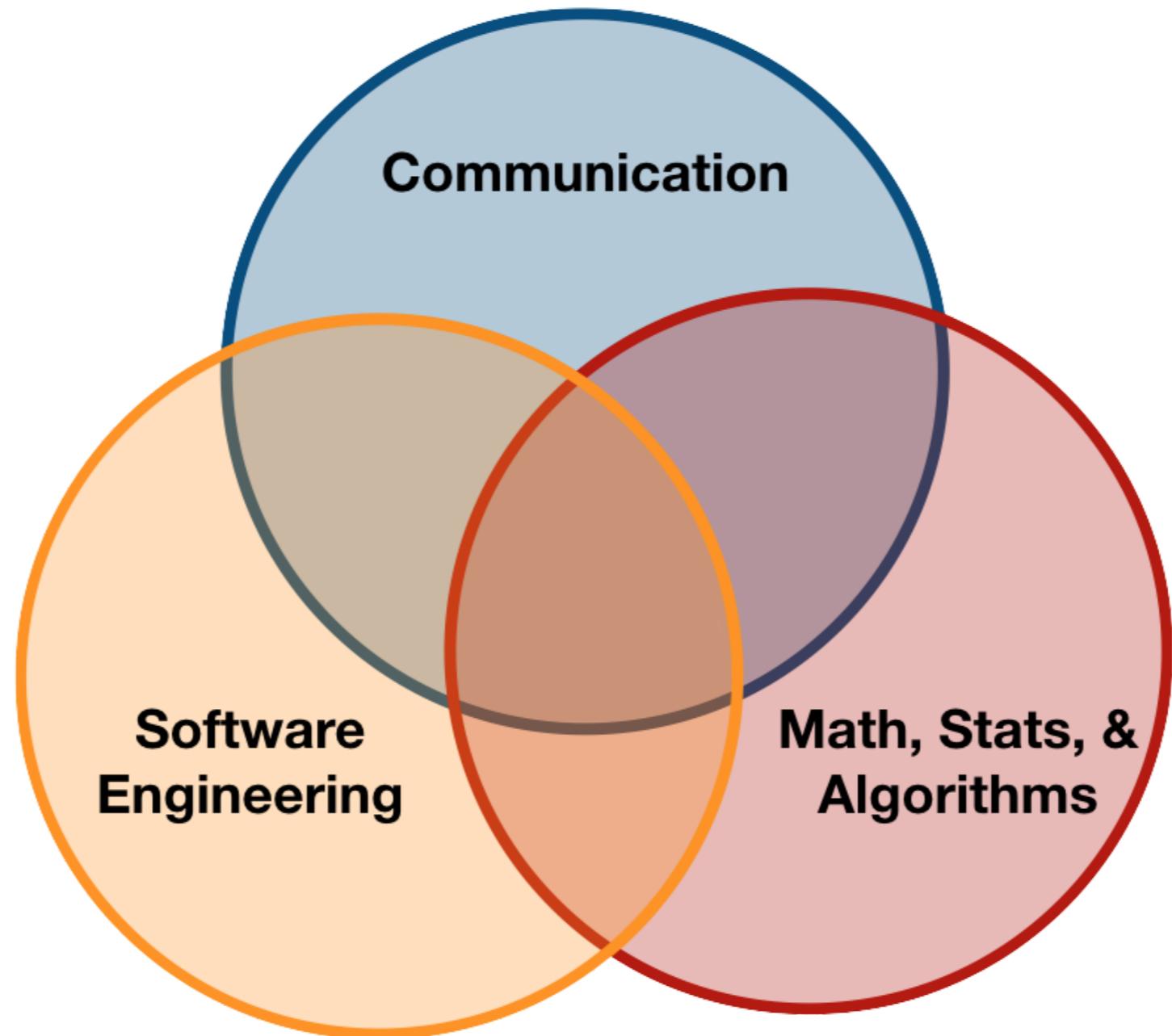
Adam Spannbauer

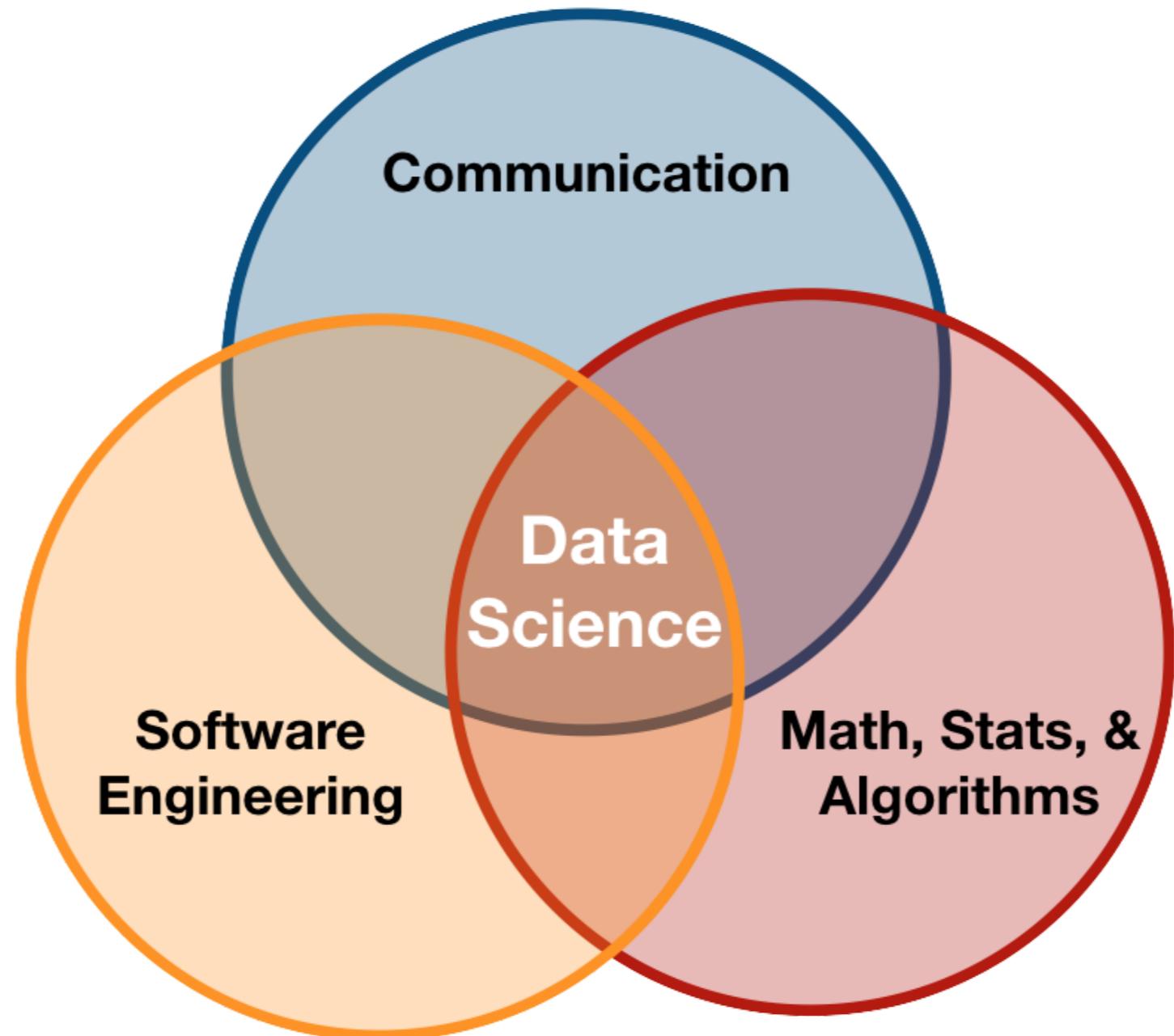
Machine Learning Engineer at Eastman









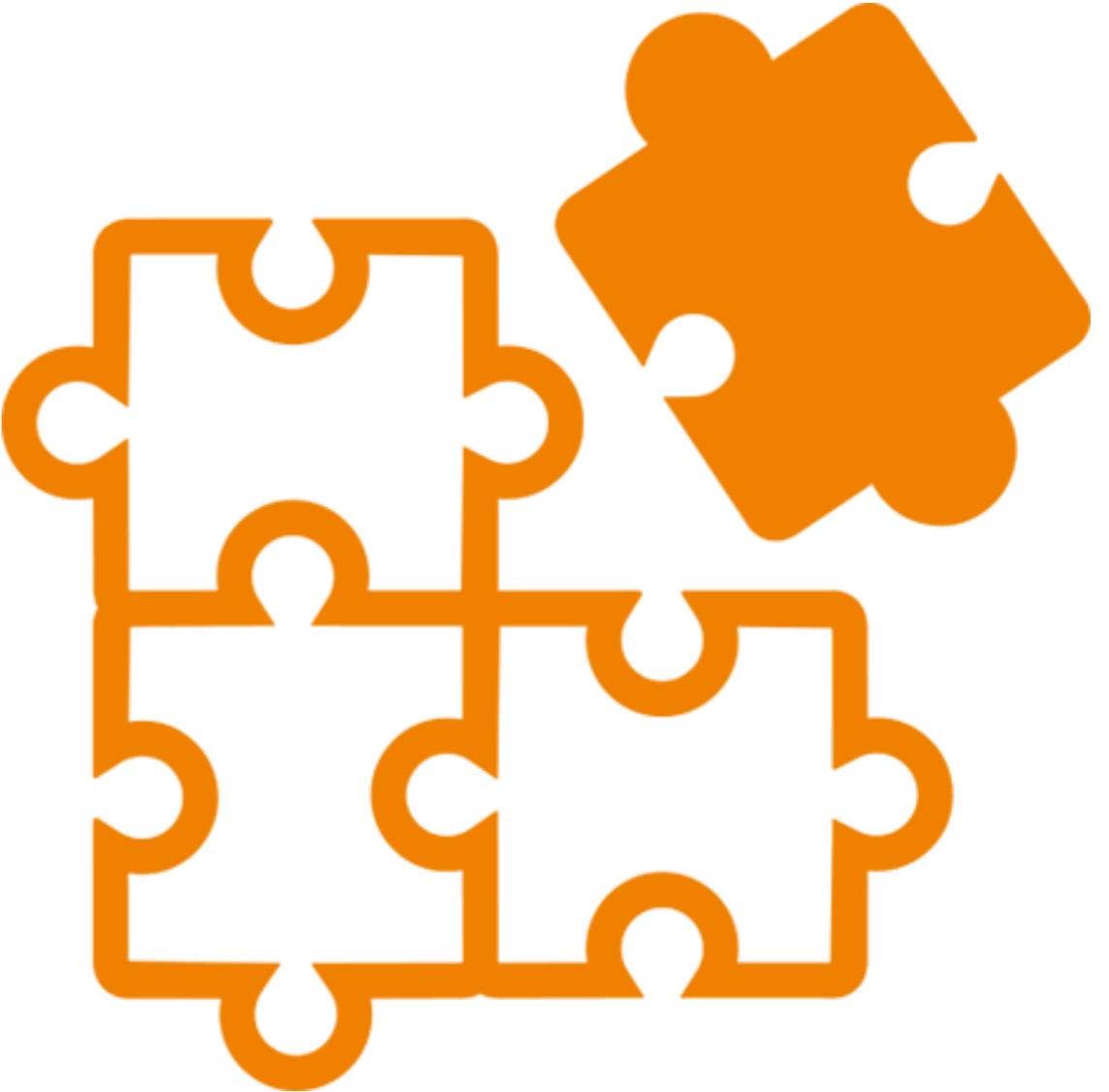


# Software engineering concepts

- Modularity
- Documentation
- Testing
- Version Control & Git

# Benefits of modularity

- Improve readability
- Improve maintainability
- Solve problems only once



# Modularity in python

```
# Import the pandas PACKAGE
import pandas as pd

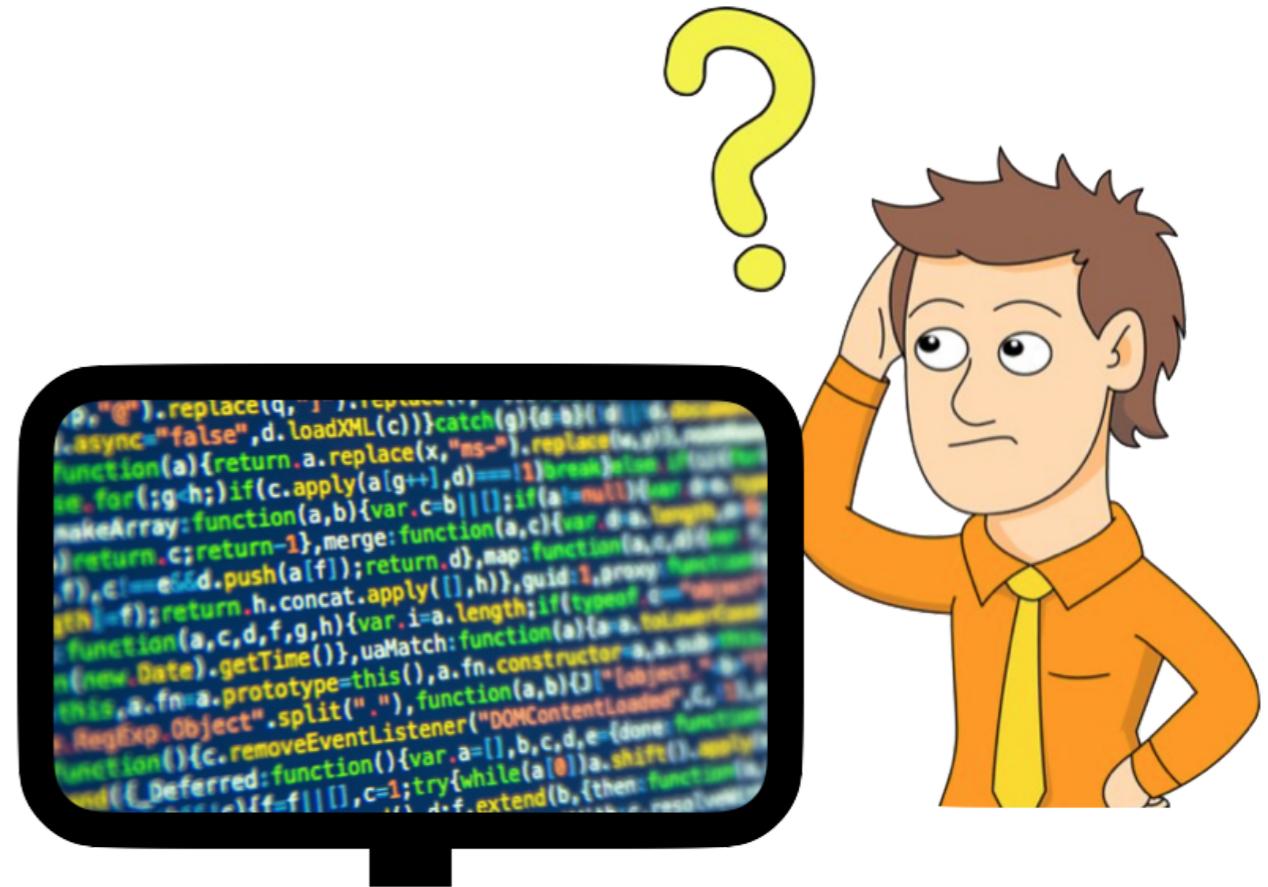
# Create some example data
data = {'x': [1, 2, 3, 4],
        'y': [20.1, 62.5, 34.8, 42.7]}

# Create a dataframe CLASS object
df = pd.DataFrame(data)

# Use the plot METHOD
df.plot('x', 'y')
```

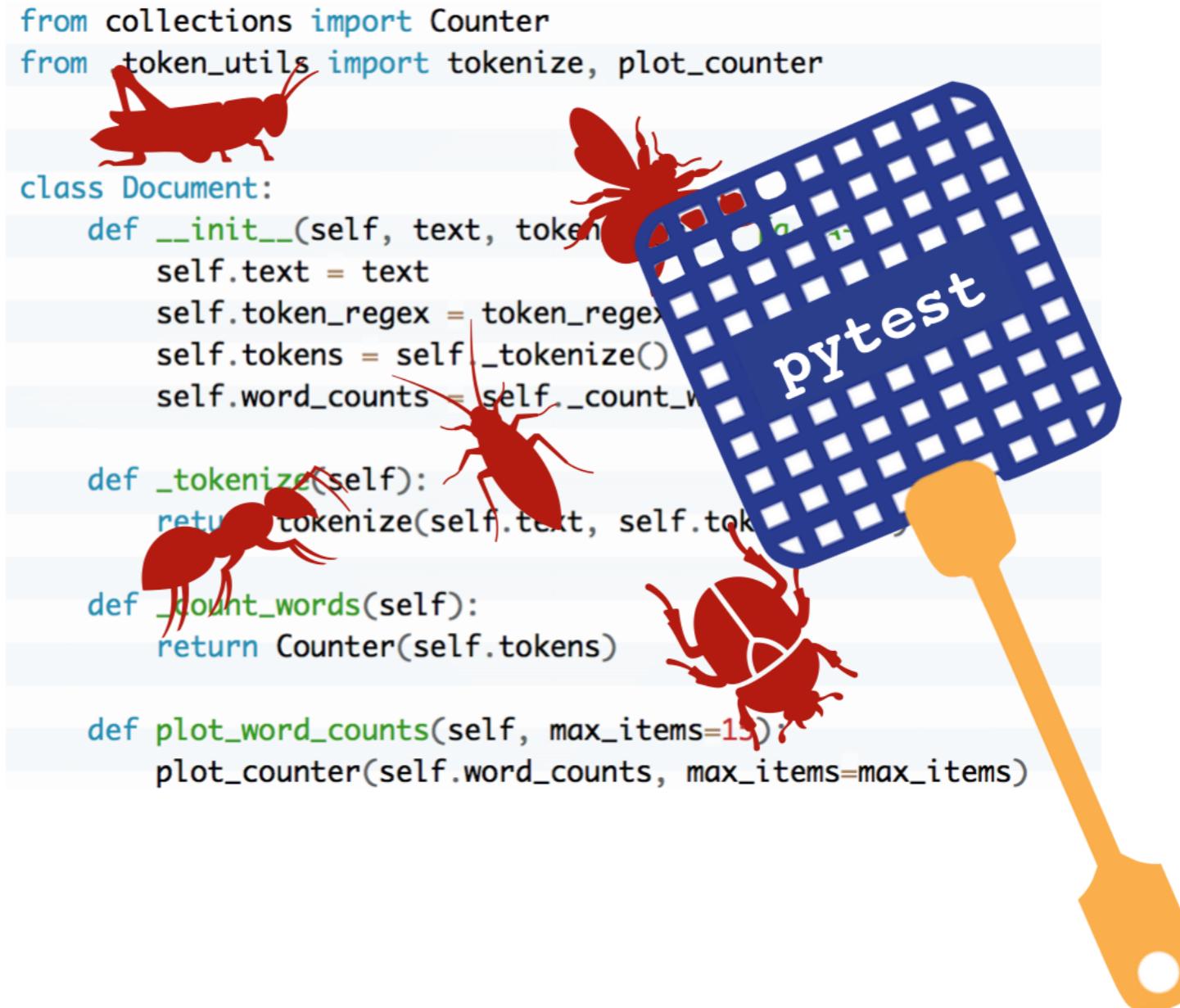
# Benefits of documentation

- Show users how to use your project
- Prevent confusion from your collaborators
- Prevent frustration from future you



# Benefits of automated testing

- Save time over manual testing
- Find & fix more bugs
- Run tests anytime/anywhere



# **Let's Review**

## **SOFTWARE ENGINEERING PRINCIPLES IN PYTHON**

# Introduction to Packages & Documentation

SOFTWARE ENGINEERING PRINCIPLES IN PYTHON

Adam Spannbauer

Machine Learning Engineer at Eastman



# Packages and PyPi



# Intro to pip



# Intro to pip



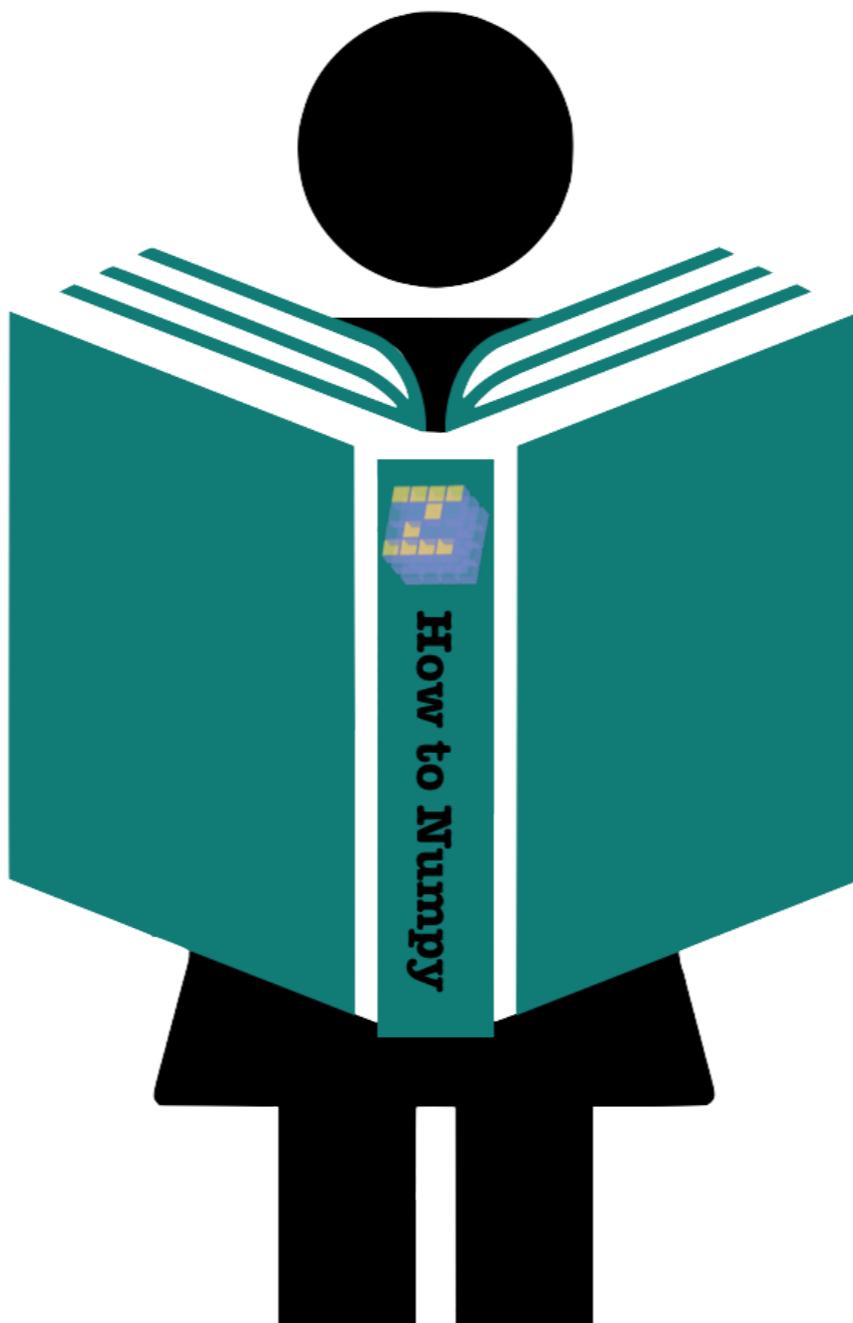
# Using pip to install numpy

```
datacamp@server:~$ pip install numpy
```

```
Collecting numpy
  100% |██████████| 24.5MB 44kB/s

Installing collected packages: numpy
Successfully installed numpy-1.15.4
```

# How do we use numpy?



# Reading documentation with help()

```
help(numpy.busday_count)
```

busday\_count(begindates, enddates)

Counts the number of valid days between `begindates` and `enddates`, not including the day of `enddates`.

Parameters

-----

begindates : the first dates for counting.

enddates : the end dates for counting (excluded from the count)

Returns

-----

out : the number of valid days between the begin and end dates.

Examples

-----

```
>>> # Number of weekdays in 2011  
... np.busday_count('2011', '2012')
```

260

# Reading documentation with help()

```
import numpy as np  
help(np)
```

Provides

1. An array object of arbitrary homogeneous items
2. Fast mathematical operations over arrays
3. Linear Algebra, Fourier Transforms, Random Number Generation

```
help(42)
```

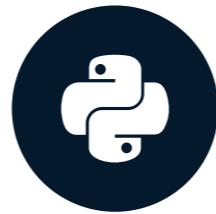
```
class int(object)  
| int(x=0) -> integer  
| int(x, base=10) -> integer  
|  
| Convert a number or string to an integer, or return 0 if no arguments  
| are given. If x is a number, return x.__int__(). For floating point  
| numbers, this truncates towards zero.
```

# **Let's Practice**

## **SOFTWARE ENGINEERING PRINCIPLES IN PYTHON**

# Conventions and PEP 8

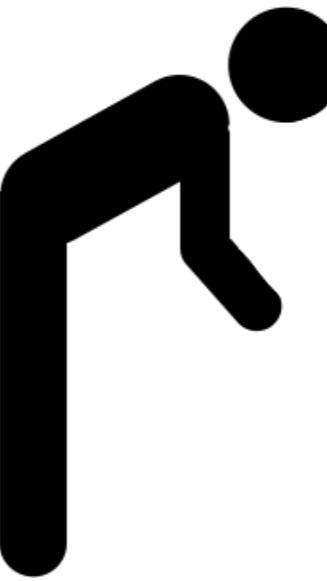
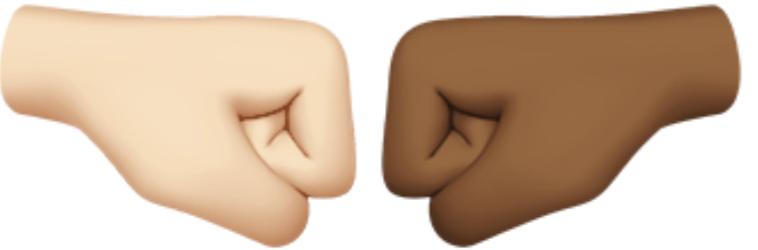
SOFTWARE ENGINEERING PRINCIPLES IN PYTHON



Adam Spannbauer

Machine Learning Engineer at Eastman

# What are conventions?



# Introducing PEP 8



"Code is read much more often than it is written"

# Violating PEP 8

```
#define our data
my_dict ={
    'a' : 10,
    'b': 3,
    'c' : 4,
    'd': 7}
#import needed package
import numpy as np
#helper function
def DictToArray(d):
    """Convert dictionary values to numpy array"""
    #extract values and convert
    x=np.array(d.values())
    return x
print(DictToArray(my_dict))
```

```
array([10,  4,  3,  7])
```

# Following PEP 8

```
# Import needed package
import numpy as np

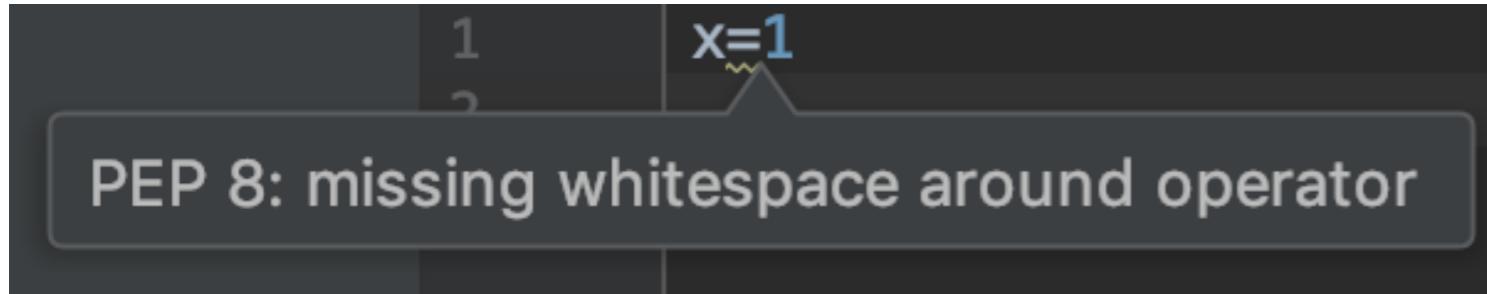
# Define our data
my_dict = {'a': 10, 'b': 3, 'c': 4, 'd': 7}

# Helper function
def dict_to_array(d):
    """Convert dictionary values to numpy array"""
    # Extract values and convert
    x = np.array(d.values())
    return x

print(dict_to_array(my_dict))
```

```
array([10,  4,  3,  7])
```

# PEP 8 Tools



# Using pycodestyle

```
datacamp@server:~$ pip install pycodestyle  
datacamp@server:~$ pycodestyle dict_to_array.py
```

```
dict_to_array.py:5:9: E203 whitespace before ':'  
dict_to_array.py:6:14: E131 continuation line unaligned for hanging indent  
dict_to_array.py:8:1: E265 block comment should start with '# '  
dict_to_array.py:9:1: E402 module level import not at top of file  
dict_to_array.py:11:1: E302 expected 2 blank lines, found 0  
dict_to_array.py:13:15: E111 indentation is not a multiple of four
```

# Output from pycodestyle

dict\_to\_array.py:9:1: E402 module level import not at top of file

The diagram illustrates the structure of a pycodestyle error message. The message is: `dict_to_array.py:9:1: E402 module level import not at top of file`. Annotations with arrows point to specific parts of the message:

- File**: Points to the file name `dict_to_array.py`.
- Line number**: Points to the line number `9`.
- Column number**: Points to the column number `1`.
- Error code**: Points to the error code `E402`.
- Error description**: Points to the descriptive text `module level import not at top of file`.

# **Let's Practice**

## **SOFTWARE ENGINEERING PRINCIPLES IN PYTHON**