Introduction to Programming for Data Science

Kylie A. Bemis

Northeastern University
Khoury College of Computer Sciences

Northeastern University

Basics

- DS 5010 Introduction to Programming for Data Science
- Schedule: Tuesdays 6:00 pm - 9:15 pm (Boston)
- Location: Knowles Center 010 + Zoom meetings
- Dates: Jan 18, 2022 - May 6, 2022
- Course website:
  - https://kuwisdelu.github.io/ds5010-spring22.html
  - Link to all relevant resources available on Canvas/Piazza

Intros: Kylie Ariel Bemis

- PhD Statistics, Purdue University
- Research includes:
  - Bioinformatics (mass spectrometry imaging)
  - Statistical computing languages (R development)
- Software (R packages):
  - "Cardinal" - mass spectrometry imaging tools
  - "matter" - out-of-memory computing
- 2015 John M. Chambers Statistical Software award for development of "Cardinal" package
- Published author of short science fiction and fantasy

Goals for today

- Course expectations and policies
- Tools and resources
- Introduction to Python for DS

COURSE EXPECTATIONS AND POLICIES

Course materials

- Syllabus:
  - https://kuwisdelu.github.io/ds5010-spring22.html
- Piazza:
  - https://piazza.com/northeastern/spring2022/ds5010bemis
- Canvas:
  - https://northeastern.instructure.com/courses/103307

Zoom meetings

- Classes will be hosted synchronously via **Zoom**
  - Meeting links for each class can be found on Canvas
  - Meetings will be recorded and made available after class (*tech willing*)
- Synchronous attendance is encouraged
  - Asking questions during class is welcome and encouraged
  - Quizzes will have a limited time frame to be completed online
- In-person attendance is not required

Announcements, questions, and queries

- Outside office hours, all course-related questions and correspondence should use **Piazza**
  - https://piazza.com/northeastern/spring2022/ds5010bemis
  - Course material and office hours schedules posted under "Resources"
  - Use "Q&A" to ask asynchronous questions to everyone or instructors only
  - Send a private note for course-related queries
- Do not email for course-related queries — it may get lost
- How to ask a good question:
  - https://stackoverflow.com/help/how-to-ask

Office hours

- Virtual office hours will be held using **Microsoft Teams**
  - Download at https://teams.northeastern.edu/
  - Live online during our scheduled office hours (see Piazza)
  - Chat or video call
- Times available on Piazza Resources "Staff" tab
- Or schedule a video call by appointment
- In-person appointments only if *absolutely necessary*

Assignments and grading

- Assignments and grading will use **Canvas**
  - https://northeastern.instructure.com/courses/103307
  - Tentative semester schedule (*subject to change*)
  - Assignments must be submitted via Canvas *by the deadline on Canvas*
  - Grades will be distributed via Canvas
- Do not email assignments — they will be ignored

Grade breakdown

- Homework: 30%
- Quizzes: 30%
- Project: 30%
- Participation: 10%

Homework

- 4 individual homework assignments
- Submit on Canvas
  - Required format to be described in each assignment
  - Grading both automated + manual
- Due every 1-2 weeks

Quizzes

- 2 "in-class" quizzes on Canvas
- Complete online during time window
- Test your conceptual understanding
- Will replace a class meeting

Project

- Teams of 2-3 students
- Complete a programming term project
- Submit a mid-semester project proposal
- Submit code repository + written report
- Detailed guidelines will be shared on Piazza and discussed in class

Participation

- Peer review another team's code repository
- Receive feedback on your own code
- Rubrics will be provided on Piazza and discussed in class

Grading policies

- Late assignments will not be graded
- Extensions may be given on a case-by-case basis
  - Request at least 48 hours ahead of deadline
  - Require a reasonable justification
- Petitions for re-grades must be made via Piazza
  - No later than 1 week after receiving the original grade
  - Clearly explain why solution is correct and should be re-graded
  - Before petitioning me, first contact the grader to understand grade
  - *The new grade may be lower than the original grade*

General policies

- Academic integrity policy
  - http://www.northeastern.edu/osccr/academic-integrity-policy/
  - Please remember you are here to learn, not manufacture good grades
  - Use work with permission; cite it properly; do not plagiarize
- Title IX policy
  - https://www.northeastern.edu/ouec/
  - Northeastern is committed to preventing discrimination and harassment
  - You may talk to me, but I am a mandatory Title IX reporter
  - Confidential resources are available if you need

General policies (cont'd)

- Please be kind and respectful to each other
  - Use other people's requested names and pronouns
  - Be considerate of other people's identities and backgrounds
  - Keep the classroom a safe and healthy environment
- Getting help
  - Please reach out as early as possible if you are struggling
  - I am less able to make accommodations closer to deadlines
  - Northeastern's WeCare office is a resource in times of stress
    - https://studentlife.northeastern.edu/we-care/

COVID-19 policies

- This course is taught using the hybrid NUflex model
  - Remote classroom participation via Zoom meetings
  - This class is intended to be a *synchronous* classroom experience
- To attend class in-person, you **must**
  - Wear a mask at all times
  - Practice social distancing
- ***Do not come to campus if you are experiencing symptoms***
- In-person classes may be cancelled if necessary

TOOLS AND RESOURCES

Python installation

- Option 1: Install from python.org
  - Official version of Python
- Option 2: Use a package manager
  - MacOS: **homebrew** (https://brew.sh/)
  - Linux: **chocolatey** (https://chocolatey.org/)
- Option 3: Use conda
  - Anaconda: https://www.anaconda.com
  - Miniconda: https://docs.conda.io/en/latest/miniconda.html

Option 1: Install from python.org

- Official version
- Widely-compatible
- Easiest and simplest way to install
- More than sufficient for this course
- *No package manager necessary*

Option 2: Use a package manager

- Platform-specific package managers:
  - MacOS: **homebrew** (https://brew.sh/)
  - Linux: **chocolatey** (https://chocolatey.org/)
- Use to install other software (e.g., *git*)
  - Example: `brew install python3`
- No scientific computing support

Option 3: Use conda

- Conda is a package manager for DS
- Two major distributions of conda:
  - Anaconda: 1,500+ packages, large install (3+ GB)
  - Miniconda: Minimalist, install packages individually
- Great for scientific computing packages
  - Easy to install Jupyter, etc.

Installing Python packages

- Use PIP to install packages from PyPI
  - Comes with most Python installations
  - **P**IP **I**nstalls **P**ackages
- Example:
  - `pip3 install numpy`

Optional: Install IPython

- IPython stands for **interactive** Python
- IPython provides several improvements over the default Python interface
  - Syntax highlighting
  - Tab completion
  - Etc.
- Install via pip, conda, etc.

Python requirements

- Make sure you have at least:
  - Python 3 (preferably ≥ 3.9)
  - PIP for Python 3
- Example:
  - `python3 --version`
  - `pip3 --version`

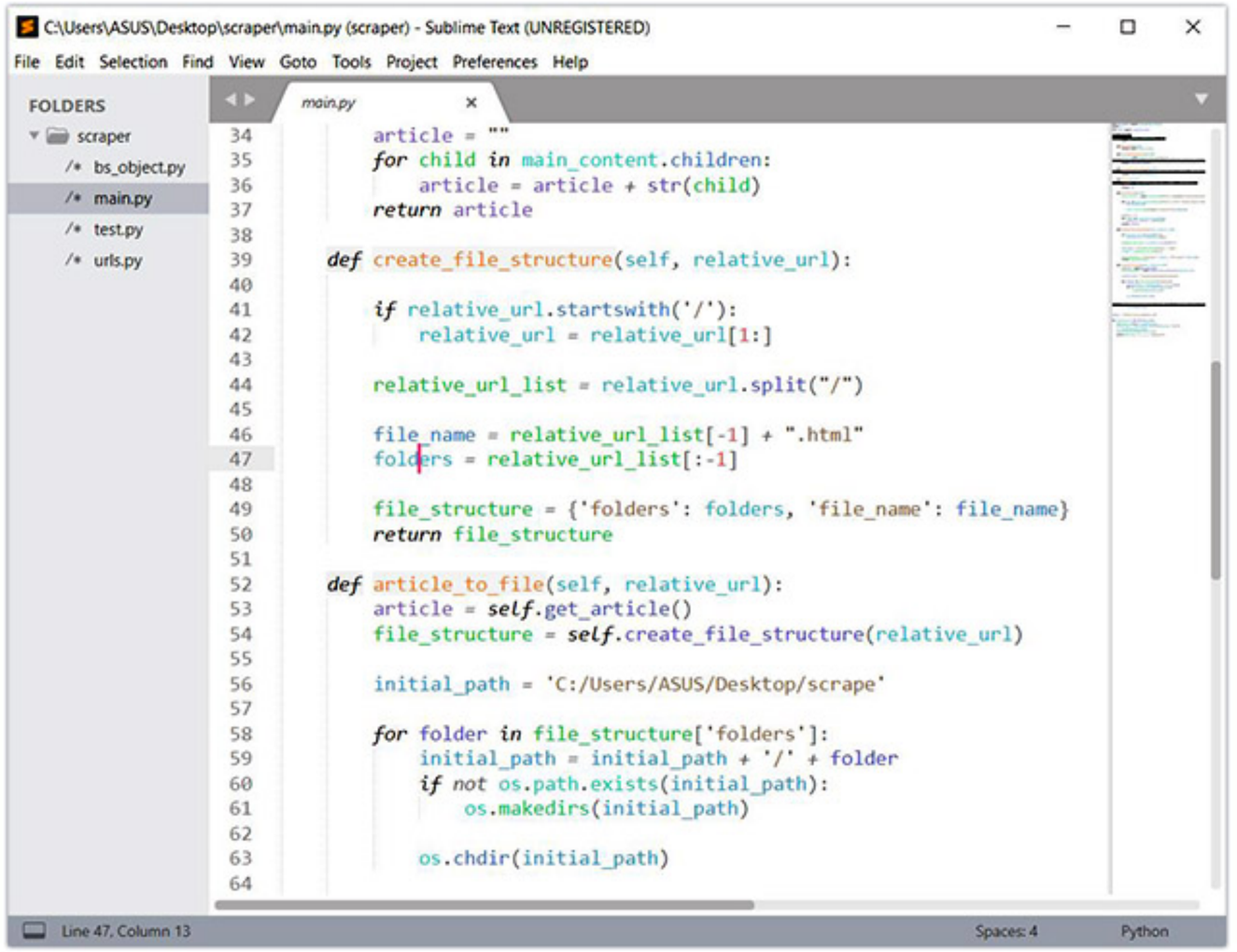
Additional installation resources

- https://docs.python-guide.org/starting/installation/
- https://www.infoworld.com/article/3530140/how-to-install-python-the-smart-way.html

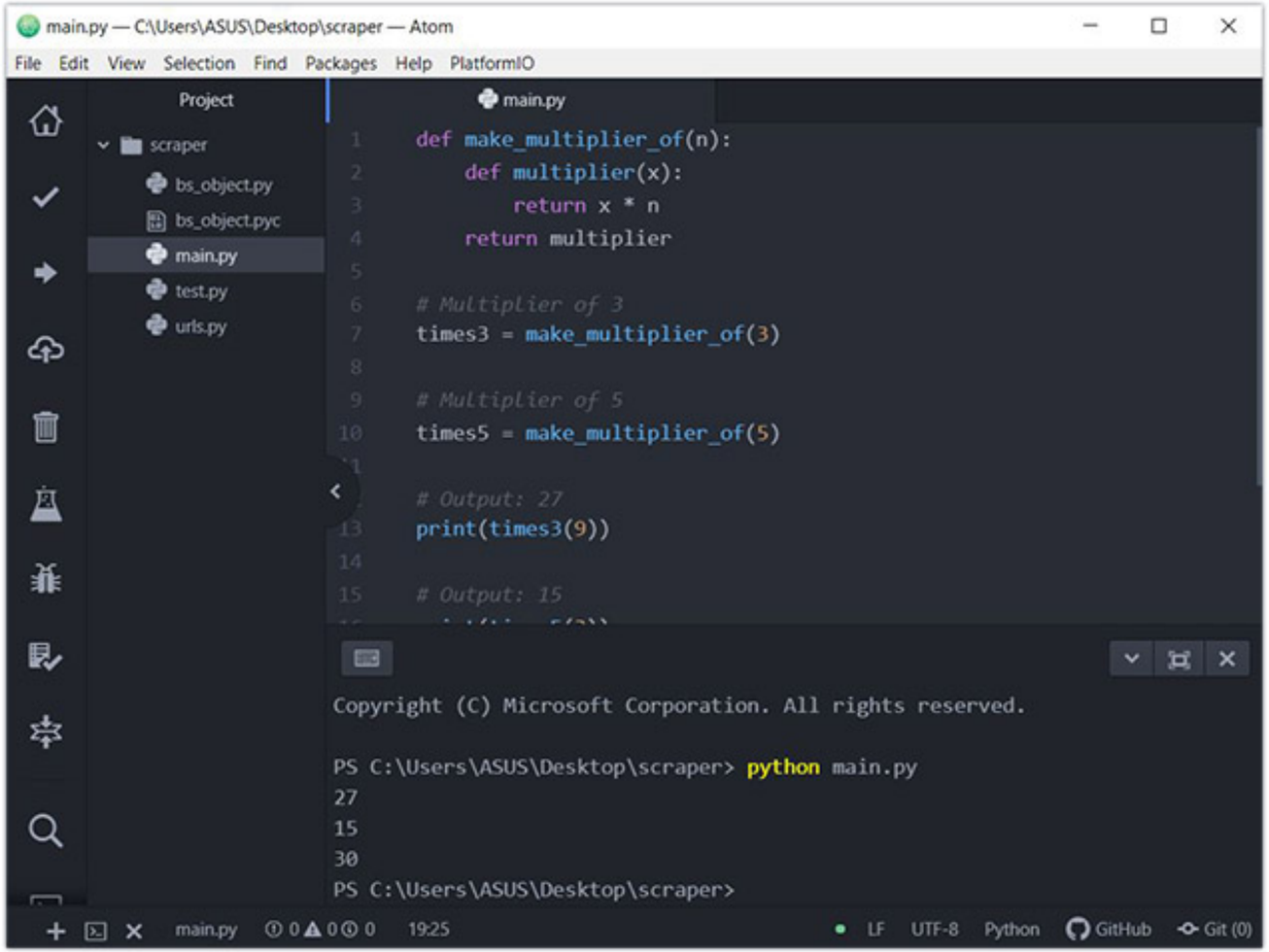
Text editors and IDEs

- Use any text editor to edit Python code
  - Sublime Text 3 (https://www.sublimetext.com)
  - Atom (https://atom.io)
  - Vim, emacs, etc.
- Spyder is a beginner-friendly IDE
- Do *not* use Jupyter (for *this* course)

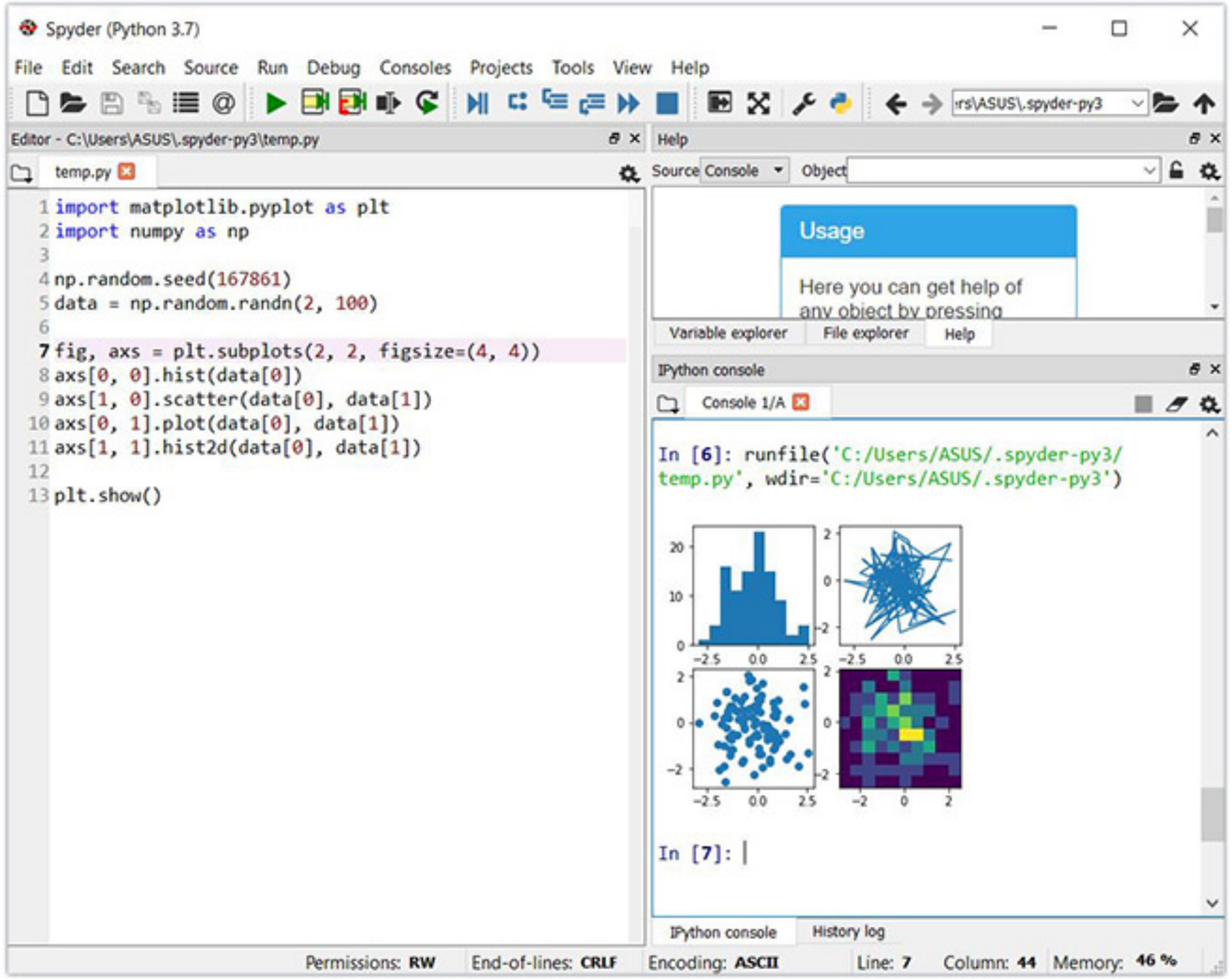
Sublime Text 3



Atom


main.py — C:\Users\ASUS\Desktop\scraper — Atom
File Edit View Selection Find Packages Help PlatformIO
Project
scraper
bs_object.py
bs_object.pyc
main.py
test.py
urls.py
main.py
def make_multiplier_of(n):
def multiplier(x):
return x * n
return multiplier
# Multiplier of 3
times3 = make_multiplier_of(3)
# Multiplier of 5
times5 = make_multiplier_of(5)
# Output: 27
print(times3(9))
# Output: 15
Copyright (C) Microsoft Corporation. All rights reserved.
PS C:\Users\ASUS\Desktop\scraper> python main.py
27
15
30
PS C:\Users\ASUS\Desktop\scraper>
main.py 0 0 0 19:25 LF UTF-8 Python GitHub Git (0)

Spyder


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - C:\Users\ASUS\.spyder-py3\temp.py
temp.py
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 np.random.seed(167861)
5 data = np.random.randn(2, 100)
6
7 fig, axs = plt.subplots(2, 2, figsize=(4, 4))
8 axs[0, 0].hist(data[0])
9 axs[1, 0].scatter(data[0], data[1])
10 axs[0, 1].plot(data[0], data[1])
11 axs[1, 1].hist2d(data[0], data[1])
12
13 plt.show()
Help
Source Console Object
Usage
Here you can get help of
any object by pressing
Variable explorer File explorer Help
IPython console
Console 1/A
In [6]: runfile('C:/Users/ASUS/.spyder-py3/
temp.py', wdir='C:/Users/ASUS/.spyder-py3')
In [7]:
IPython console History log
Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 7 Column: 44 Memory: 46 %

Textbooks

- All required reading sections will be available on Piazza
- *Learning Python.* Mark Lutz. O'Reilly Media, 2013.
- *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.* Wes McKinney, O'Reilly Media, 2013.
- *Introduction to Algorithms, 3rd edition.* Thomas Cormen, Charles Leiserson, Ronald Rivest and Clifford Stein, MIT Press, 2009.

How to ask for help

- Write titles that summarize the problem
  - Bad: "Doubt on homework"
  - Good: "Why does my function return float instead of int?"
- Explain the problem
  - What is the problem you're trying to solve?
  - What have you tried so far to fix it?
- https://stackoverflow.com/help/how-to-ask

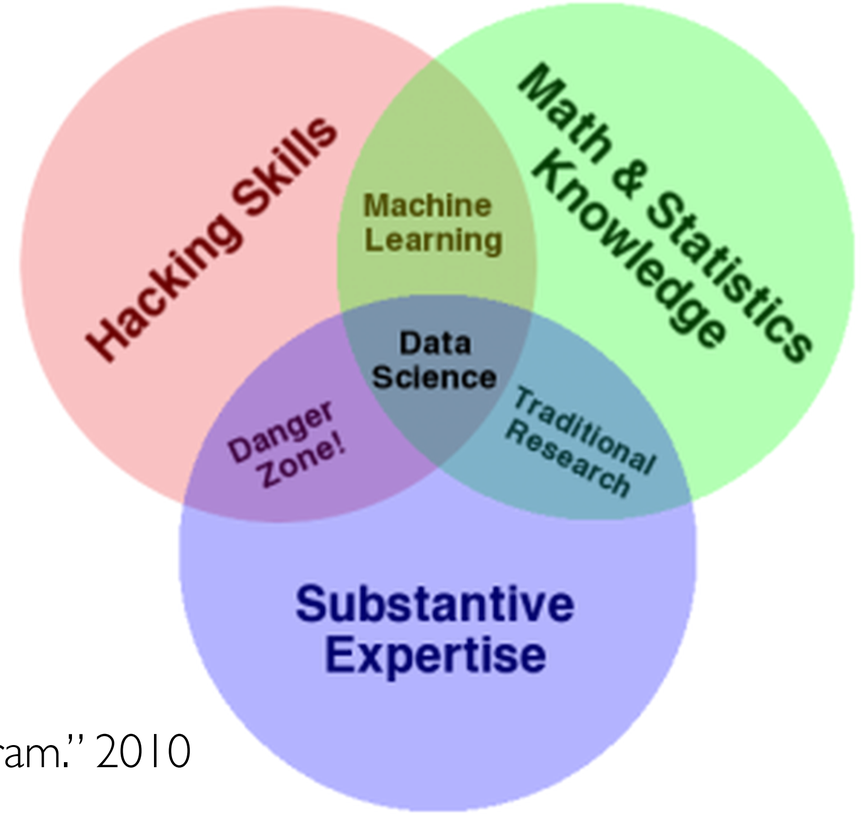
How to ask for help (2)

- Provide a reproducible example (reprex)
  - Provide the minimum amount of code to reproduce
  - Make sure your example reproduces your problem
  - Ideally, the example should run self-contained
  - Provide the actual code, and *not just a screenshot*
- https://stackoverflow.com/help/minimal-reproducible-example

INTRODUCTION TO **PYTHON** FOR DATA SCIENCE

What is data science?

- A **data scientist** blends several different areas of expertise to draw insights from data
  - Statistics and machine learning
  - Computer science and programming
  - Domain knowledge
- This course will focus on:
  - Programming (i.e., the "hacking" skills)



Drew Conway. "The Data Science Venn diagram." 2010

Why programming?

- Programming is a vital skill for data scientists
- GUI-based tools are limited
- Necessary for data analysis to be:
  - Reproducible
  - Deployable

Why Python?


SQL


mongoDB


hadoop


Google Cloud


R


python


K
Keras


aws


Google BigQuery


TensorFlow


APACHE
Spark


django

Why Python?

- Python is the glue of data science
  - Many projects require multiple languages + tools
  - Python interfaces with most APIs
- Python is a capable DS toolkit
  - NumPy + Pandas
  - scikit-learn + statsmodels
  - matplotlib

A brief history of Python

- Developed by Guido van Rossum
  - First released in 1991
  - "Benevolent Dictator for Life" (until 2018)
- Designed to be easy and fun to use
- Compatible across many platforms
- Widely adopted by the data science community

Python is…

- High-level
  - Strong abstraction from details of computer hardware
- Interpreted
  - Does not need to be formally compiled to be run
- General purpose
  - Widely applicable across many domains

Using the Python REPL

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 1
2
>>> 2 * 3
6
```

- Python provides a REPL for interactive use:
  - **R**ead-**E**val-**P**rint-**L**oop
- Type statements into the Python REPL
  - Statements are interpreted and evaluated after hitting "Enter"

Example: "Hello world"

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, world!")
Hello, world!
>>>
```

`print()` is a function

Basic data types

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> type(1)
<class ‘int’>
>>> type(1.)
<class ‘float’>
>>> type(None)
<class ‘NoneType’>
>>> type(True)
<class ‘bool’>
```

- Python provides several basic data types:
- **bool, int, float, str, NoneType**

Use indentation to structure code

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> for i in range(3):
...     print(i)
0
1
2
>>>
```

- Python uses **indentation** to structure code
- Use either tabs or spaces—be consistent!

Getting help

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> help(range)

Help on built-in function len in module builtins:

len(obj, /)
    Return the number of items in a container.
```

- Use `help()` to access documentation
- Also available through the pydoc module

Exiting the Python REPL

```
Python 3.9.1 (default, Jan  8 2021, 17:17:17)
[Clang 12.0.0 (clang-1200.0.32.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
%
```

- Use the `exit()` function to exit the REPL

Exercises at home

- Download and install Python
- Start Python and evaluate test code:
  - print("Hello, world!")
  - 1 + 2
  - help(print)
- Choose a text editor or IDE to use