ODSC EUROPE 2021

BASIC PYTHON FOR DATA PROCESSING

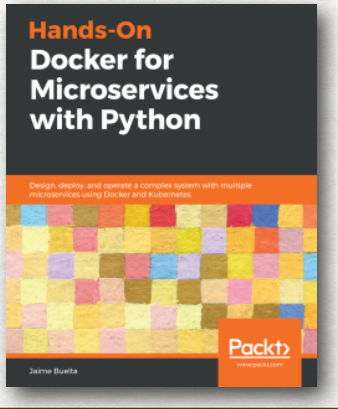
JAIME BUELTA

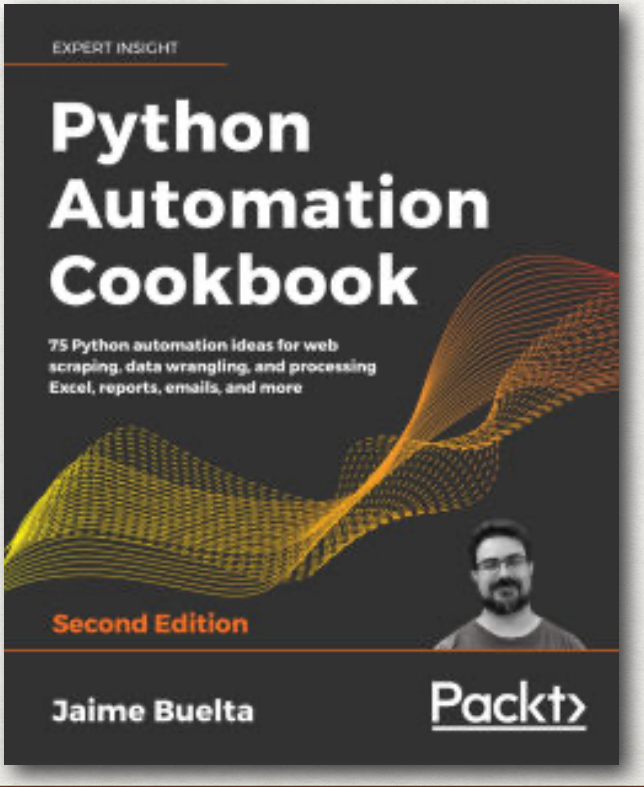
ABOUT ME

- Software Architect in Double Yard
- Python developer since 2010
- Book author (Amazon)
- Podcast wrongsideoflife.com

buelta.com

@jaimebuelta





AIM OF THE WORKSHOP

- Get an introduction of the Python possibilities
- Start a journey in useful elements for dealing with data
  - Data input from files
  - Data output to files
  - Useful third-party modules like Matplotlib, Pandas, sh…

SUMMARY

1. Basic Python
2. Dealing with files
3. Efficient data treatment

PRERREQUISITES

INSTALL PYTHON 3

Likely already installed! Check running

- https://www.python.org/downloads/

WHY USING PYTHON

- Easy to use. Easy to create prototype and experiment
- Multilanguage support
- Powerful standard library
- Fantastic third-party library
- Integrated environments

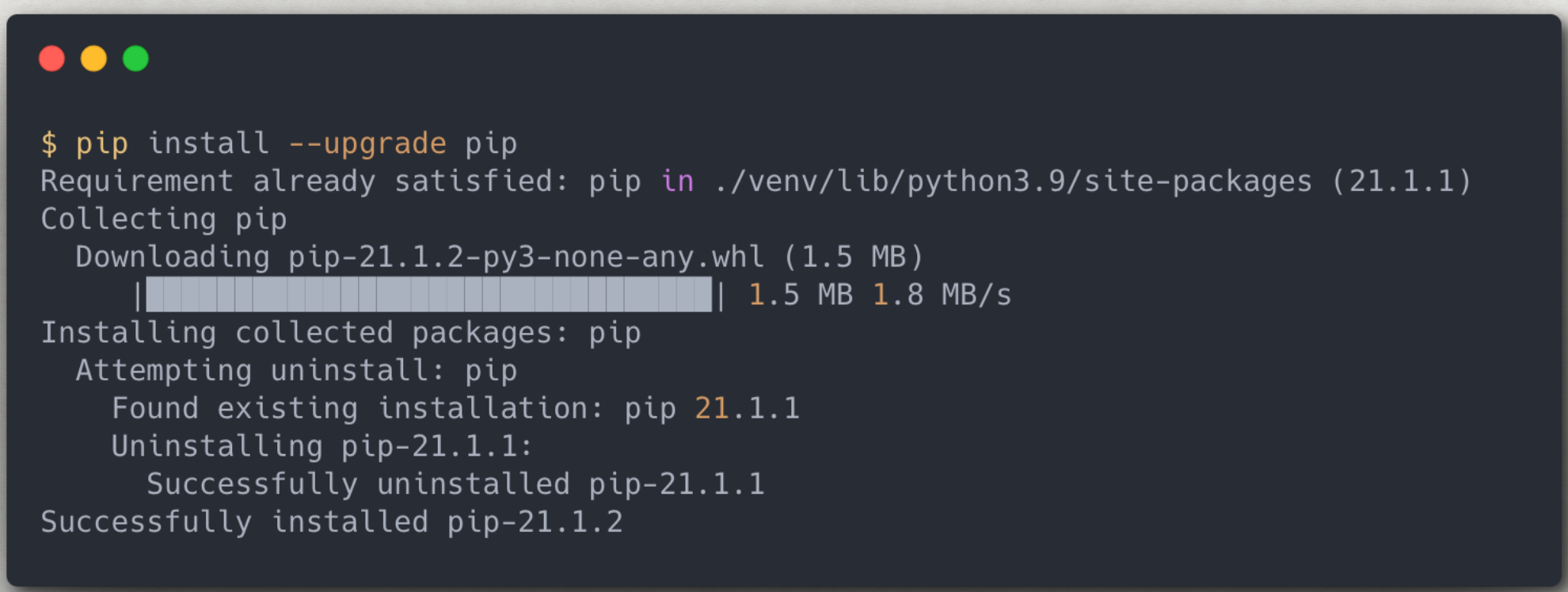
GITHUB REPO

https://github.com/jaimebuelta/odsc_2021_python_workshop

BASIC PYTHON

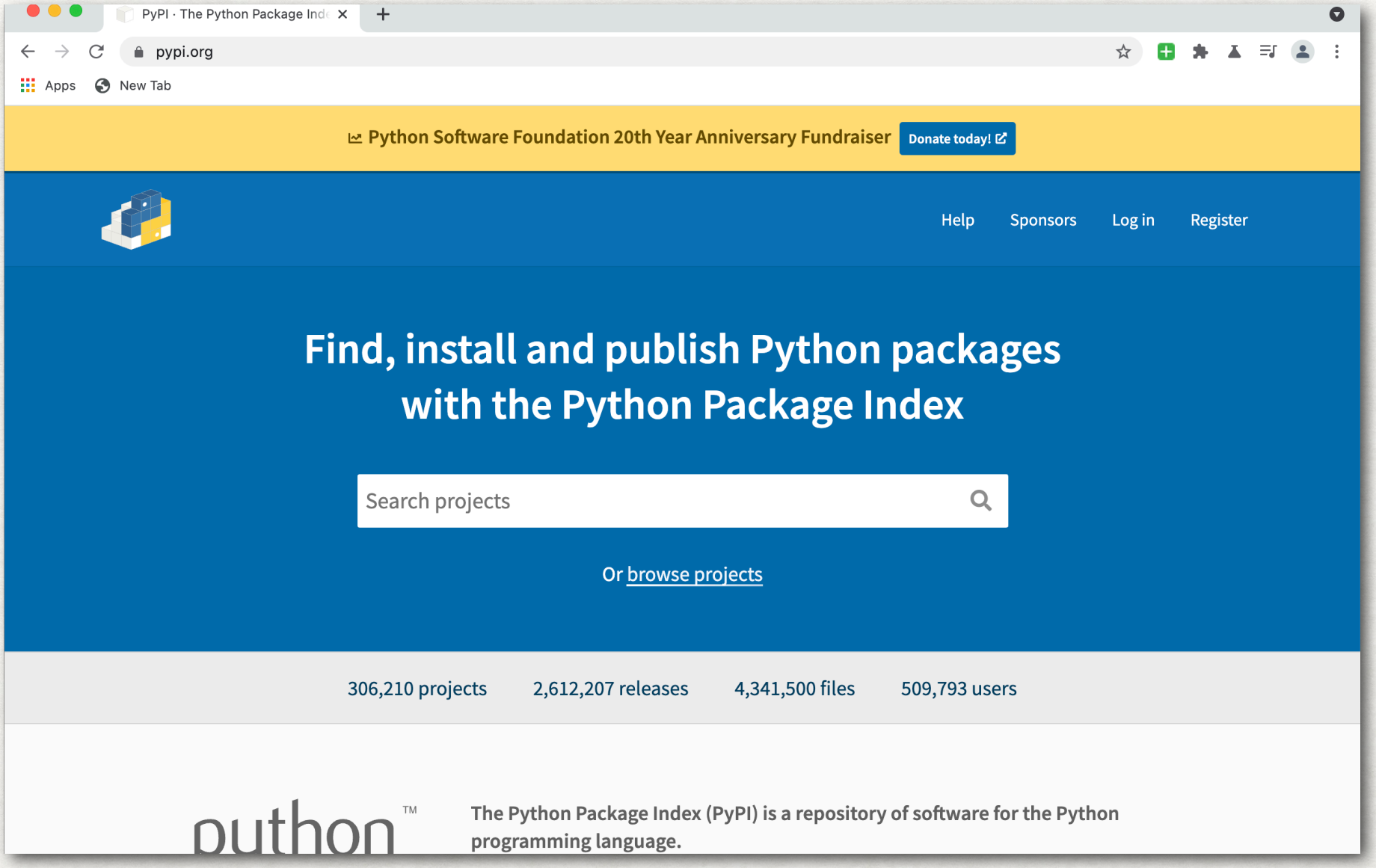
VIRTUAL ENVIRONMENTS

PIP


$ pip install --upgrade pip
Requirement already satisfied: pip in ./venv/lib/python3.9/site-packages (21.1.1)
Collecting pip
Downloading pip-21.1.2-py3-none-any.whl (1.5 MB)
| 1.5 MB 1.8 MB/s
Installing collected packages: pip
Attempting uninstall: pip
Found existing installation: pip 21.1.1
Uninstalling pip-21.1.1:
Successfully uninstalled pip-21.1.1
Successfully installed pip-21.1.2

PYPI

HTTPS://PYPI.ORG/



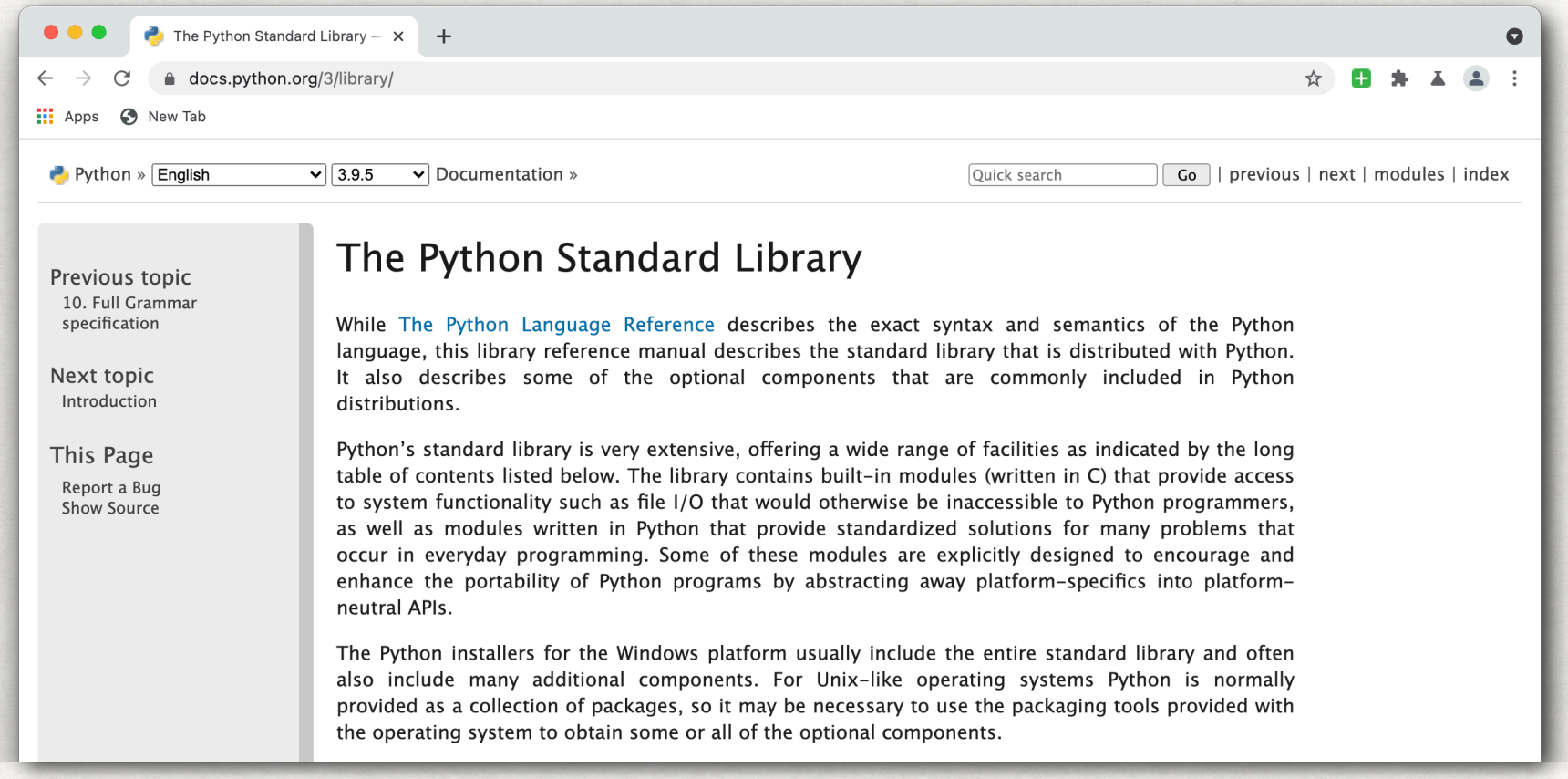
LIST OF EXTERNAL PACKAGES WE WILL USE

- click (https://pypi.org/project/click8/)
  CLI interfaces
- sh (https://pypi.org/project/sh/)
  Call external programs
- Pandas (https://pypi.org/project/pandas/) Numerical treatment
- Matplotlib (https://pypi.org/project/matplotlib/)
  Draw graphics
- FPDF (https://pypi.org/project/fpdf/)
  Create PDF files
- python-docx (https://pypi.org/project/python-docx/)
  Create .docx files
- mistune (https://pypi.org/project/mistune/)
  Render Markdown

OTHER INTERESTING PACKAGES

- Requests (to do HTTP requests)
- Django (Web framework)
- Pytest (Tests framework)
- NumPy
- SciPy
- Delorean (dealing with time and date)
- Pint (Work with physical units)

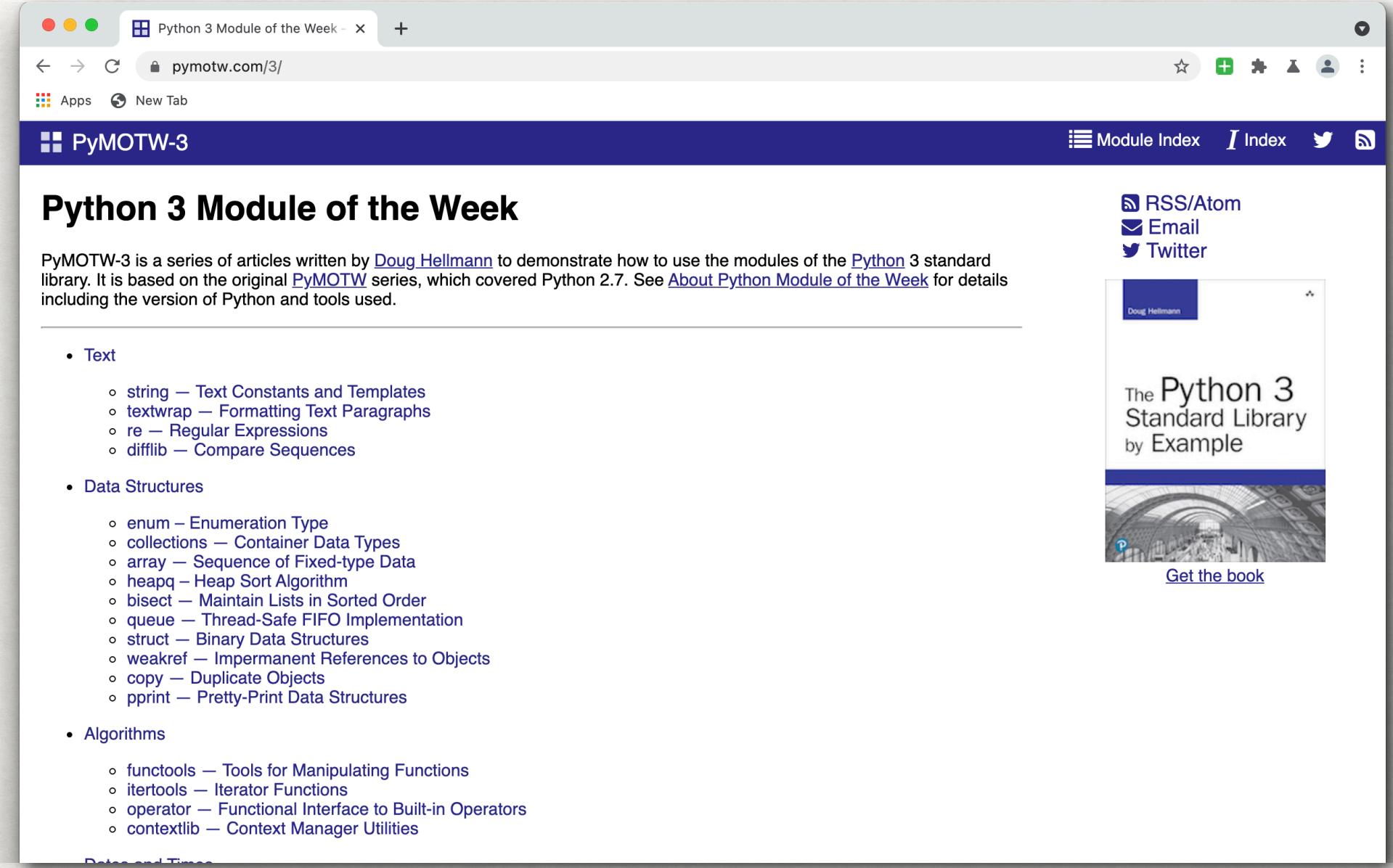
OTHER INTERESTING PACKAGES



Batteries included!

PYTHON MODULE OF THE WEEK

PYMOTW.COM



INSTALLING MULTIPLE FILES

```
requirements.txt
---
click
sh
pandas
matplotlib
pint
delorean
```

```
$ pip install -r 01_basic_python/requirements.txt
Collecting sh
  Downloading sh-1.14.2-py2.py3-none-any.whl (40 kB)
     |████████████████████████████████| 40 kB 1.8 MB/s
Collecting pandas
...
Installing collected packages
$
```

CLI INTERFACES

CLI_BASICS.PY

```python
import click
import time


@click.command()
@click.argument('number', type=int)
@click.option('--text', default='EXAMPLE')
def main(number, text):
    for _ in range(number):
        click.echo(click.style('echo ', fg='green') + text)
        time.sleep(.2)


if __name__ == '__main__':
    main()
```

EXERCISE 01

CREATE A CLI SCRIPT THAT

- Input two parameters:
  - Trigonometric function (sin, cos, tan)
  - Number between 0 and 360

- Returns the value

```
$ python3 exercise_01.py sin 0
SIN(0) = 0.0
$ python3 exercise_01.py sin 90
SIN(90) = 1.0
$ python3 exercise_01.py tan 0
TAN(0) = 0.0
$ python3 exercise_01.py cos 0
COS(0) = 1.0
```

Tip: Check Python library `math`

BASIC TYPES

```
int = 1
float = 0.3
bool = True | False
string = "some text"
None

# Groups
dict = {'key': value, 'key2': 'value2' }
list = [value, value2, value]
tuple = (value, value2, value)
set = {value, value2}
```

ADDRESSING ELEMENTS

```
list[2] = value
tuple[1] == value
value, value2 = tuple


'key' in dict
dict['key']
dict['key'] = value
del dict['key']


value in set
set.add(value)
set.remove(value)
```

ITERATING THROUGH GROUP TYPES

```python
for i in list:
  print(i)

for i in tuple:
  print(i)

for key in dict:
  print(key)

for key, value in dict.items():
  print(key, value)

for i in set:
  print(i)
```

CALL OTHER PROCESSES

SH

```
import sh


# awk -F "\"*,\"*" '{print $1}' simpsons_data.csv | uniq -c

result = sh.uniq(sh.awk('-F', '\"*,\"*', '{print $1}', 'simpsons_data.csv'),
                 '-c')
seasons = {}

for line in result:
    if 'Season' in line:
        continue

    number, season = line.split()
    seasons[season] = int(number)

print(seasons)
print(sum(seasons.values()))
```

EXERCISE 2

USE SH TO UNCOMPRESS A FILE

```
$ python3 exercise_02.py
```

Extract zipped_file.zip

DEALING WITH FILES

OPEN FILES

```
with open(filename) as fp:
  for line in fp:
    print(line)


with open(binary_filename, 'rb') as fp:
  content = fp.read()
```

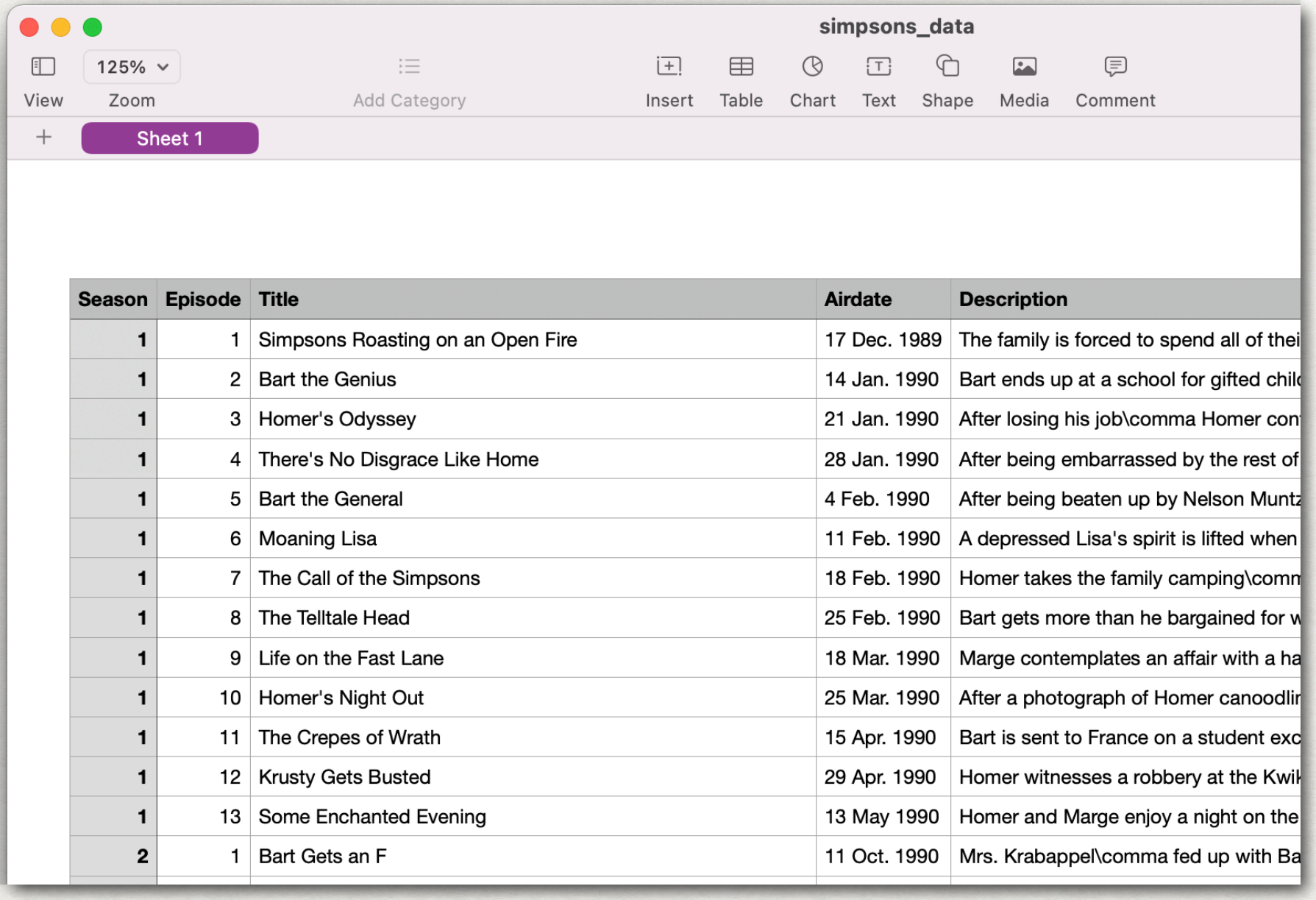
WRITE FILES

```
with open(filename, 'w') as fp:
  for line in data:
    fp.write(line + '\n')


with open(binary_filename, 'wb') as fp:
  fp.write(data)
```

ENCODINGS

```
>>> with open('text_iso.txt') as fp:
...     for l in fp:
...         print(l)
...
    ...
UnicodeDecodeError: 'utf-8' codec can't decode byte 0xa3 in
position 2: invalid start byte

>>> with open('text_iso.txt', encoding='iso-8859-1') as fp:
...     for l in fp:
...         print(l)
...
20£
```

SIMPSONS_DATA.CSV

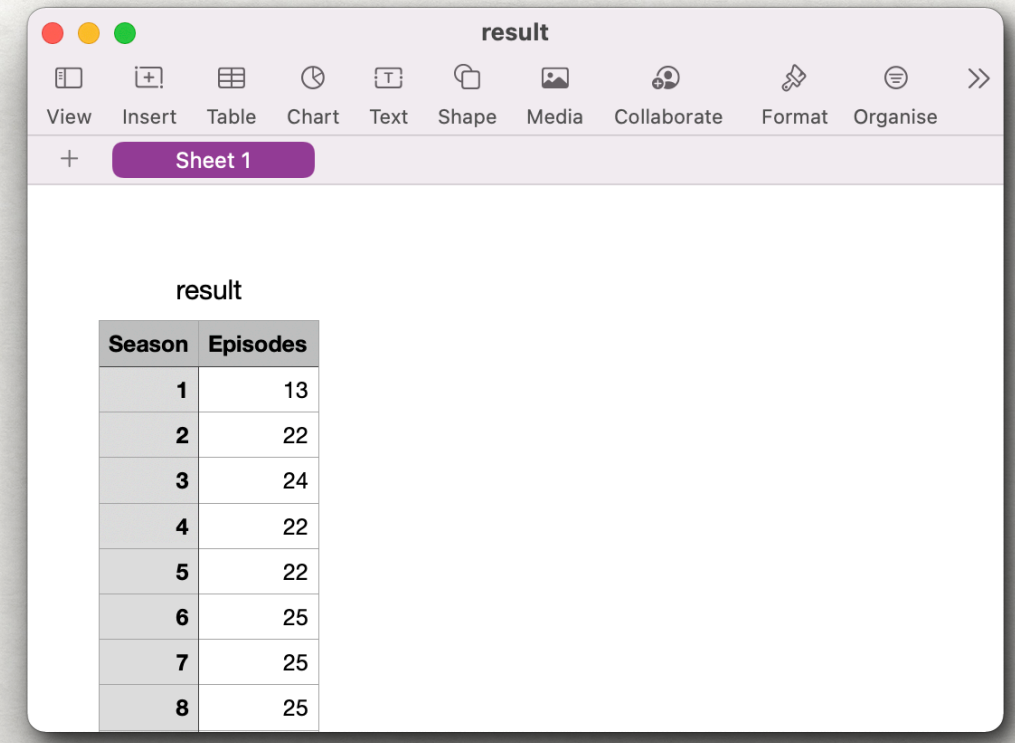


Season	Episode	Title	Airdate	Description
1	1	Simpsons Roasting on an Open Fire	17 Dec. 1989	The family is forced to spend all of thei
1	2	Bart the Genius	14 Jan. 1990	Bart ends up at a school for gifted chil
1	3	Homer's Odyssey	21 Jan. 1990	After losing his job\comma Homer con
1	4	There's No Disgrace Like Home	28 Jan. 1990	After being embarrassed by the rest of
1	5	Bart the General	4 Feb. 1990	After being beaten up by Nelson Muntz
1	6	Moaning Lisa	11 Feb. 1990	A depressed Lisa's spirit is lifted when
1	7	The Call of the Simpsons	18 Feb. 1990	Homer takes the family camping\comm
1	8	The Telltale Head	25 Feb. 1990	Bart gets more than he bargained for w
1	9	Life on the Fast Lane	18 Mar. 1990	Marge contemplates an affair with a ha
1	10	Homer's Night Out	25 Mar. 1990	After a photograph of Homer canoodlin
1	11	The Crepes of Wrath	15 Apr. 1990	Bart is sent to France on a student exc
1	12	Krusty Gets Busted	29 Apr. 1990	Homer witnesses a robbery at the Kwik
1	13	Some Enchanted Evening	13 May 1990	Homer and Marge enjoy a night on the
2	1	Bart Gets an F	11 Oct. 1990	Mrs. Krabappel\comma fed up with Ba

EXERCISE 3

Using the file `simpsons_data.csv` in the repo, create a cli script that:

- reads the file
- Determine how many episodes are per season
- Write a file with the info, in CSV format



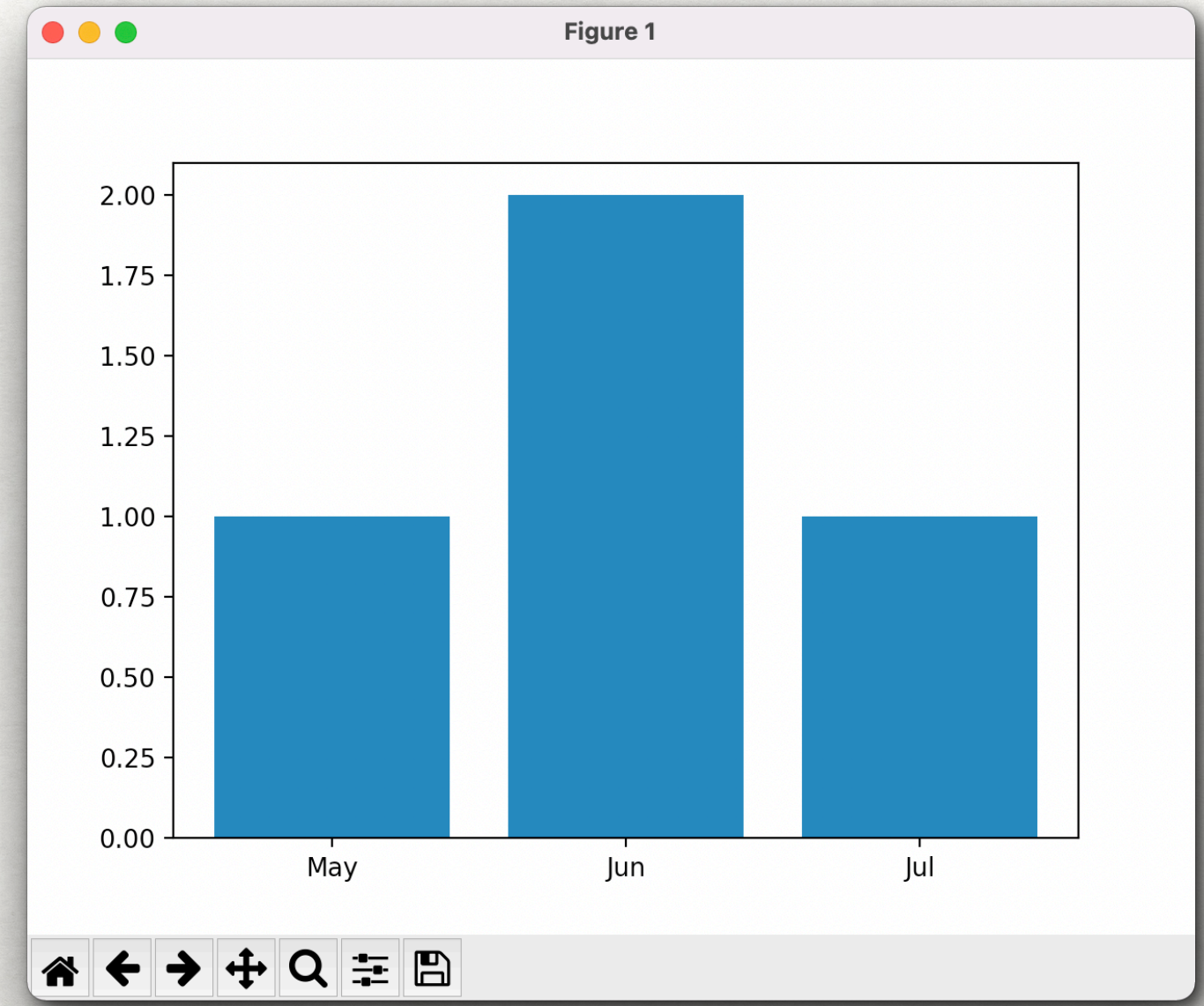
Season	Episodes
1	13
2	22
3	24
4	22
5	22
6	25
7	25
8	25

CREATE A GRAPHIC

MATPLOTLIB

```
>>> import matplotlib.pyplot as plt

>>> pos = [0, 1, 2]
>>> values = [1, 2, 1]
>>> labels = ['May', 'Jun', 'Jul']

>>> plt.bar(pos, values)
>>> plt.xticks(pos, labels)
...
>>> plt.show()
```

MATPLOTLIB



MATPLOTLIB

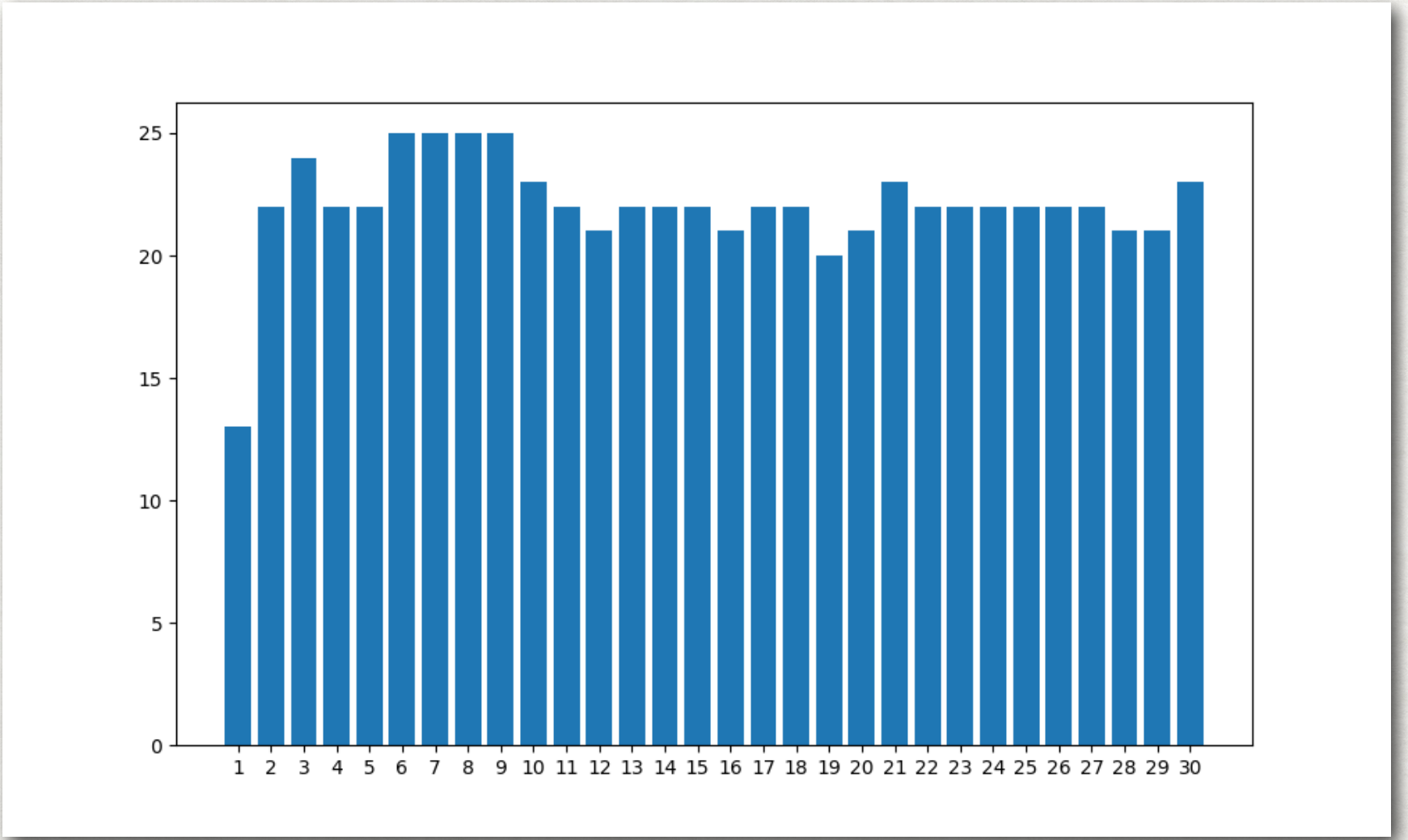
TO SAVE

```
# Adjust size
>>> figure = plt.gcf()
>>> figure.set_size_inches(5, 5)
# save
>>> plt.savefig('result.png')
```

EXERCISE 4

WRITE A PNG FILE WITH SEASON DATA

```
$ python3 exercise_04.py simpsons_data.csv result.png
```



WRITE A PDF

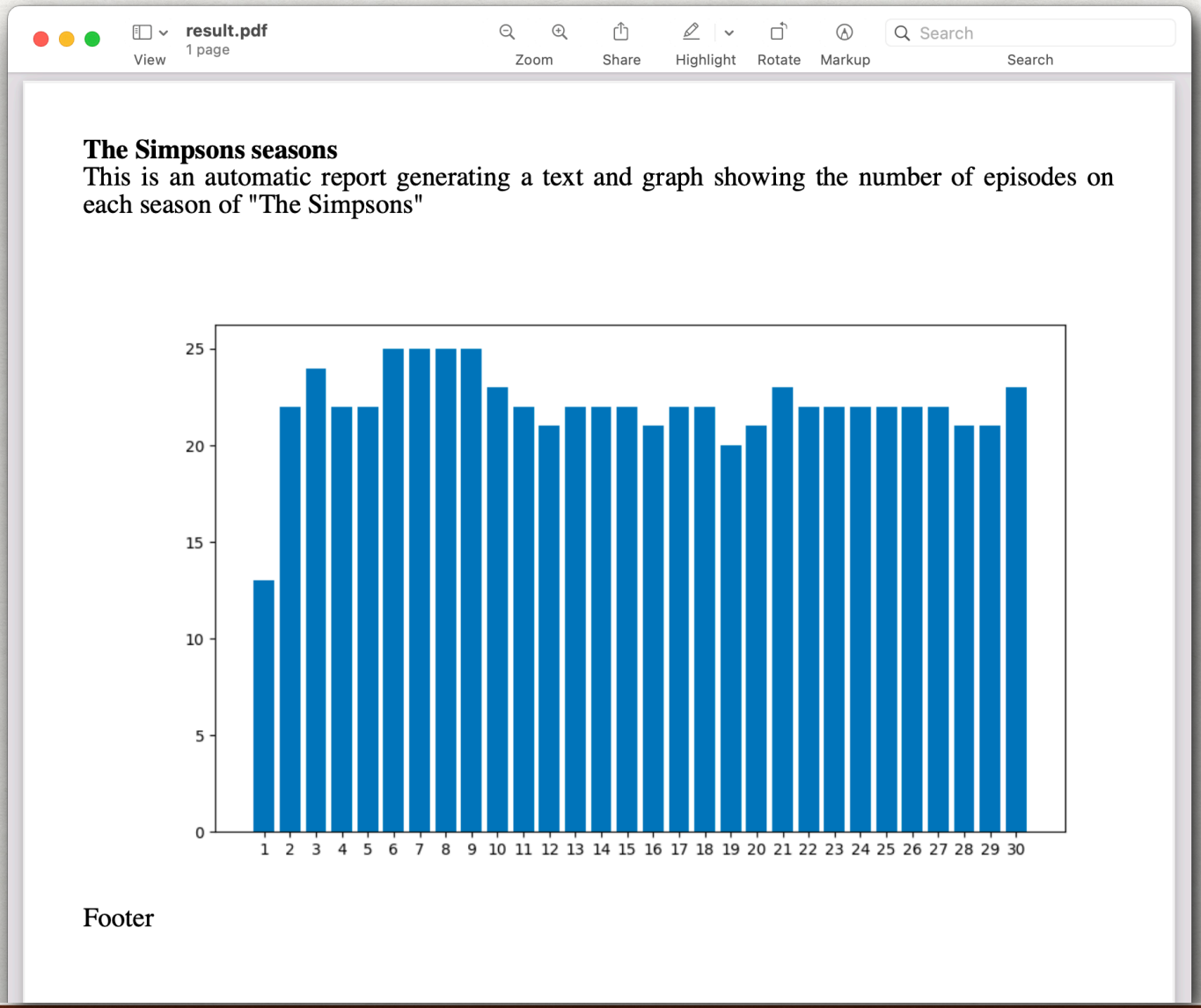
FPDF

```
document = fpdf.FPDF()
document.set_font('Times', '', 11)
document.add_page()

document.cell(0, 5, 'Example text')
document.ln()

document.output(file_out)
```

EXERCISE 5

WRITE A PDF WITH SOME TEXT AND THE GRAPH

```
$ python3 exercise_05.py simpsons_data.csv result.pdf
```

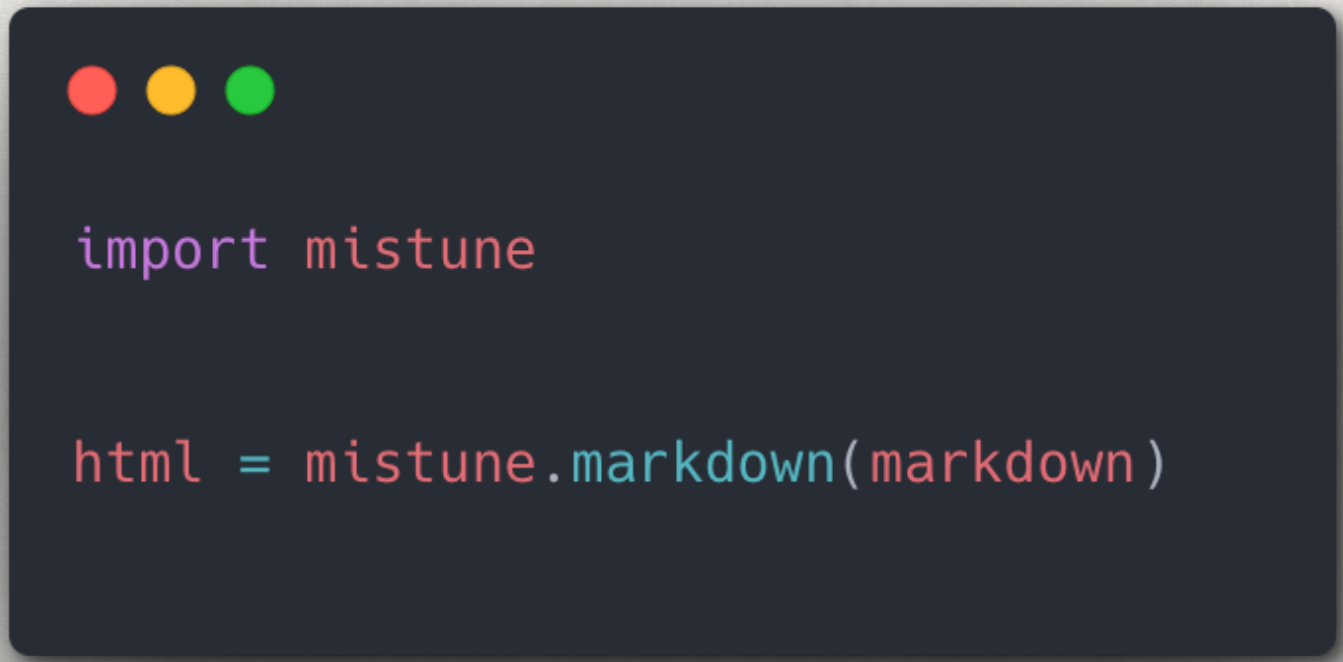


CREATE HTML TEXT

THROUGH MARKDOWN AND MISTUNE

```
Template in Markdown
---

Create a template and keep the {key} values in `.format()` way,
to transform them

```
>> TEMPLATE = 'This is text with {element}'
>> result = TEMPLATE.format(element='replaced')
>> result
'This is text with replaced'
```
```

MISTUNE

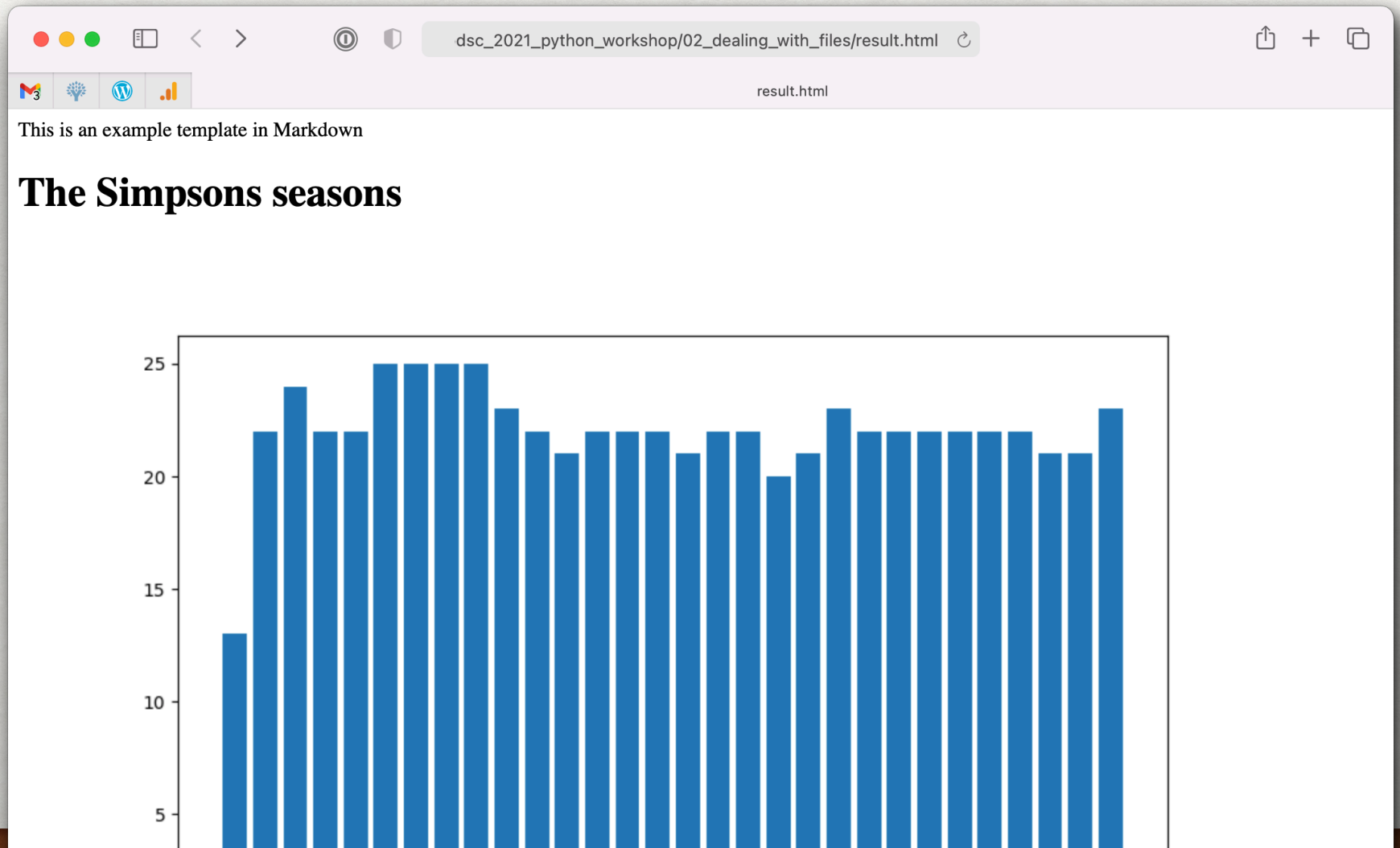


```
import mistune


html = mistune.markdown(markdown)
```

EXERCISE 6

CREATE AN HTML REPORT

```
$ python3 exercise_06.py simpsons_data.csv template.md result.html
```



CREATE WORD TEXT

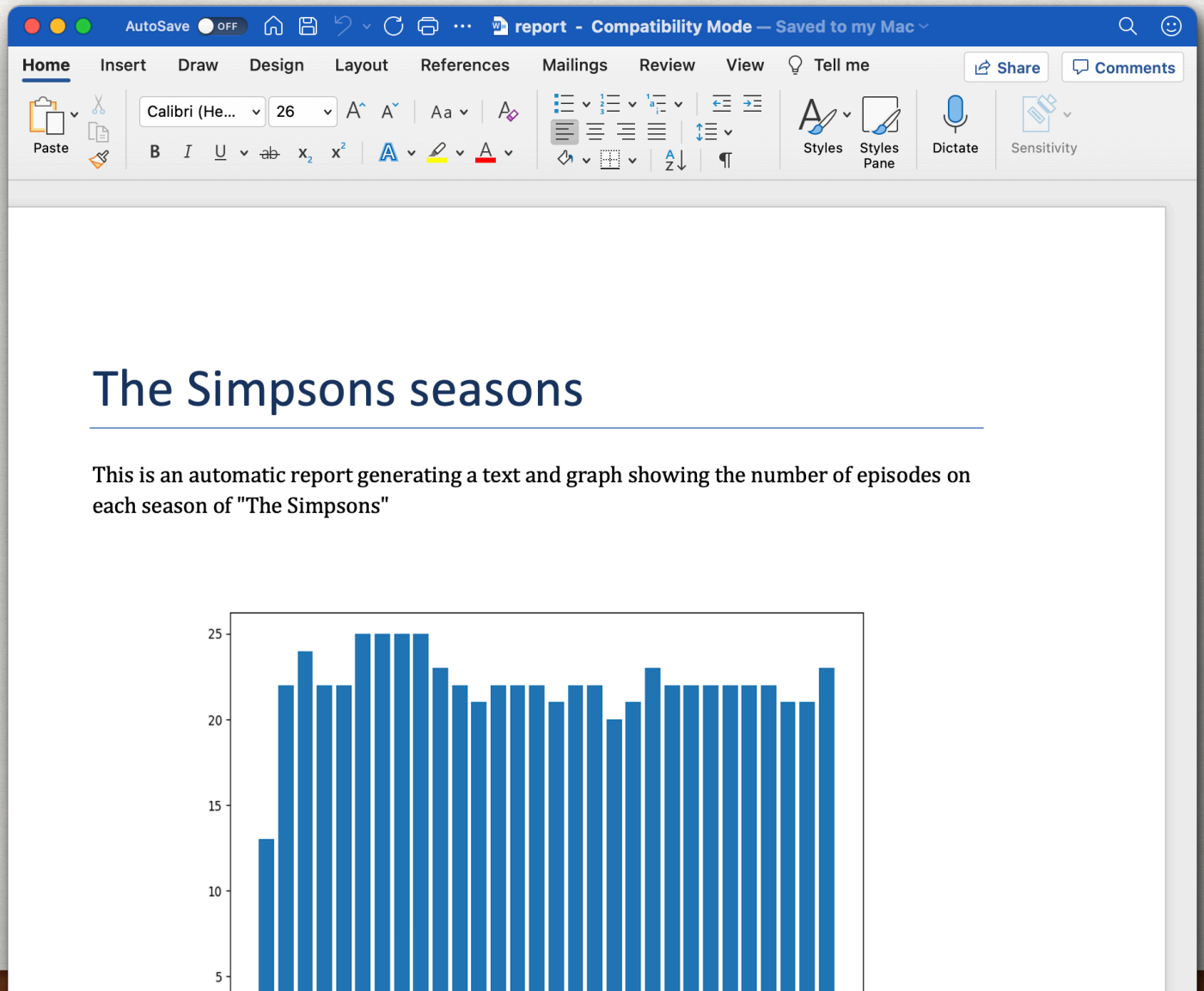
PYTHON-DOCX

```python
import docx

document = docx.Document()

paragraph = document.add_paragraph()
paragraph.add_run('Each run is a small piece of text. ')
r = paragraph.add_run('Multiple can be added and styled indepently')
r.font.name = 'Arial'

document.save('result.docx')
```

EXERCISE 7

CREATE WORD REPORT

```
$ python3 exercise_07.py simpsons_data.csv result.docv
```



EFFICIENT DATA TREATMENT

SPECIALISED DATA TYPES

- Tuples
- Array
- Deques
- Data classes

ARRAY

NUMERIC ARRAYS

```
>>> from array import array
# 'I' is for unsigned int
>>> myarray = array('I')
>>> myarray.extend([1,2,3])
>>> myarray.append('a')
...
TypeError: an integer is required (got type str)
>>> myarray.append(-1)
...
OverflowError: can't convert negative value to unsigned int
```

MUCH MORE MEMORY EFFICIENT

```
myarray = list(range(100_000_000))

myarray = array('I', range(100_000_000))
```

```
$ python3 array_memory_test.py
Current memory usage is 410.177592MB; Peak was 410.177608MB
$ python3 list_memory_test.py
Current memory usage is 3599.992996MB; Peak was 3599.992996MB
```

DEQUES

STRUCTURES OPTIMISED FOR ADDING AND REMOVING ELEMENTS ON EACH END

```
from collections import deque

myqueue = deque(range(100_000_000))
myqueue = list(range(100_000_000))

for _ in range(10):
    myqueue.append(1)
    myqueue.pop(0)      myqueue.popleft()
    len(myqueue)
```

```
$ python3 deque_appending.py
Time is 7 us
$ python3 list_appending.py
Time is 1243022 us
```

EXERCISE 8

- Moving averages using a deque, using average_values.txt

```
$ cat average_values.txt
67
45
39
...
```

```
$ python3 moving_average.py
67, 45, 39, 87, 36, 71, 61, 19, 71, 62
  ,    , 50, 57, 54, 64, 56, 50, 50, 50
```

NAMEDTUPLE

HELPS ORGANISING DATA INTO LIGHT OBJECTS

```
from collections import namedtuple


City = namedtuple('City', 'name, population, area_km2')


dublin = City(name='Dublin', population=554_554, area_km2=117.8)
toledo = City(name='Toledo', population=84_282, area_km2=232.1)
```

DATA CLASSES

USEFUL FOR OOP

```python
from dataclasses import dataclass


@dataclass
class City:
    name: str
    population: int
    area_km2: float
    capital: bool = False


dublin = City(name='Dublin', population=554_554, area_km2=117.8, capital=True)
toledo = City(name='Toledo', population=84_282, area_km2=232.1)
bad = City(name='Toledo', population='84_282', area_km2=232.1)
```

PANDAS INTRO

PANDAS_EXAMPLE.PY

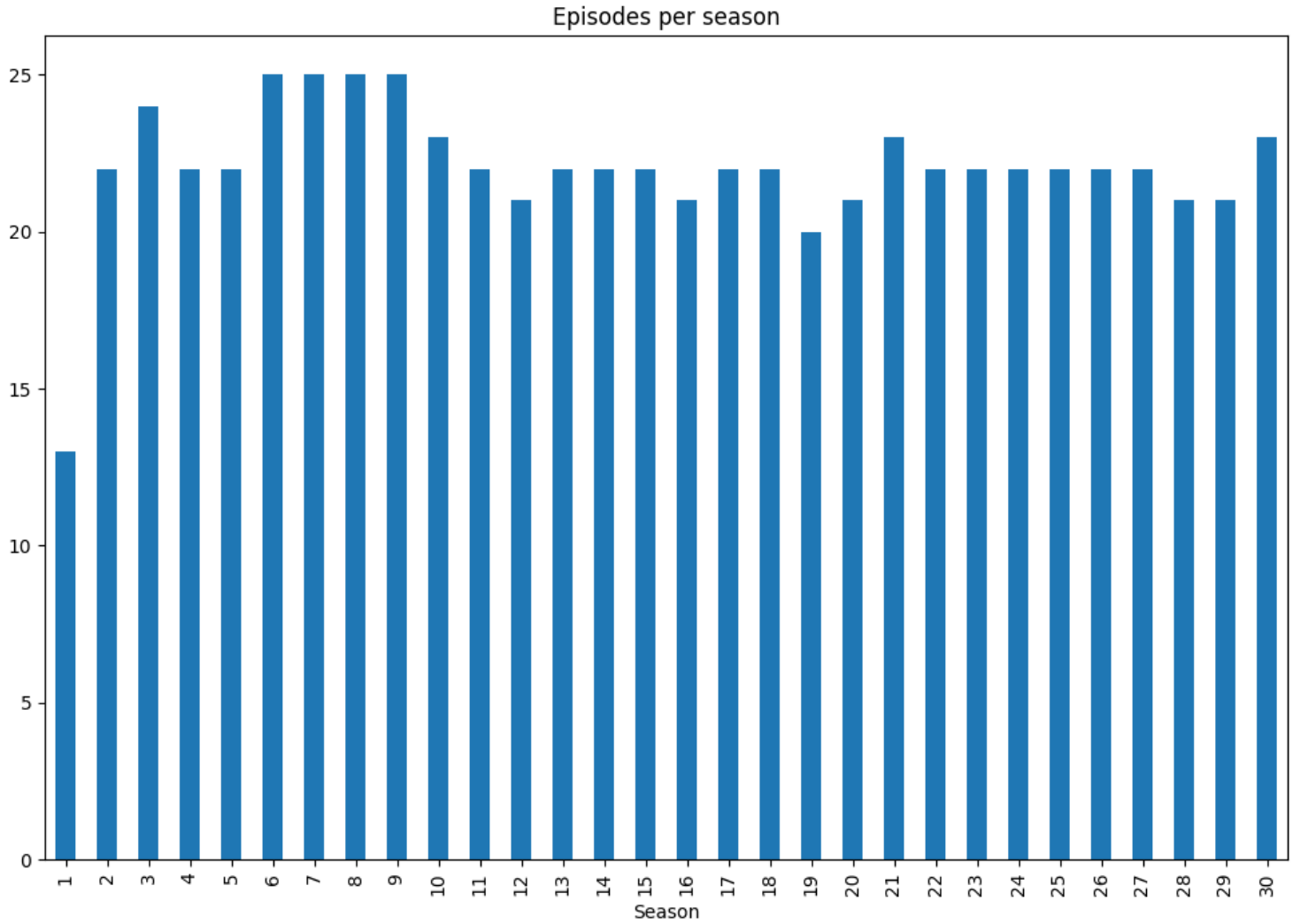
```python
import pandas as pd
from dataclasses import dataclass
import matplotlib.pyplot as plt


@dataclass
class City:
    name: str
    population: int
    area_km2: float

    def pandas(self):
        data = {
            'name': self.name,
            'population': pd.to_numeric(self.population),
            'area_km2': pd.to_numeric(self.area_km2),
        }
        return data


cities = [
    City(name='Dublin', population=554_554, area_km2=117.8),
    City(name='Toledo', population=84_282, area_km2=232.1),
]
```

PANDAS INTRO

```
data = pd.DataFrame(city.pandas() for city in cities)
print(data)
data = data.set_index('name')
print(data)

# Create a figure to store the graph
fig, axs = plt.subplots(figsize=(12, 8))

# Create a bar graph in that area
data['population'].plot.bar(ax=axs)
plt.title('Population')

# Save the result in a png
fig.savefig('cities_population.png')

# Aggregate results
result = data.agg(['sum', 'count'])
print(result)
```

EXERCISE 9

```
$ python3 seasons.py simpsons_data.csv simpsons.png
```



SPEEDING UP PYTHON WITH CYTHON

- Cython uses pyx files with pseudo Python code that can be compiled into equivalent C code.
- This speeds up significantly their execution

EXAMPLE

PRIMES.PY

```python
NUM_PRIMES = 50000


def check_if_prime(number):

    for i in range(2, number):
        if number % i == 0:
            return False

    return True


if __name__ == '__main__':
    primes = [number for number in range(2, NUM_PRIMES)
              if check_if_prime(number)]
    print(primes)
```

EXAMPLE WITH CYTHON

```
# Will compile the code the first time is imported
import pyximport
pyximport.install()

from cython_primes import check_if_prime


NUM_PRIMES = 50000


if __name__ == '__main__':
    primes = [number for number in range(2, NUM_PRIMES)
              if check_if_prime(number)]
    print(primes)
```

CYTHON CODE

cython_primes.pyx

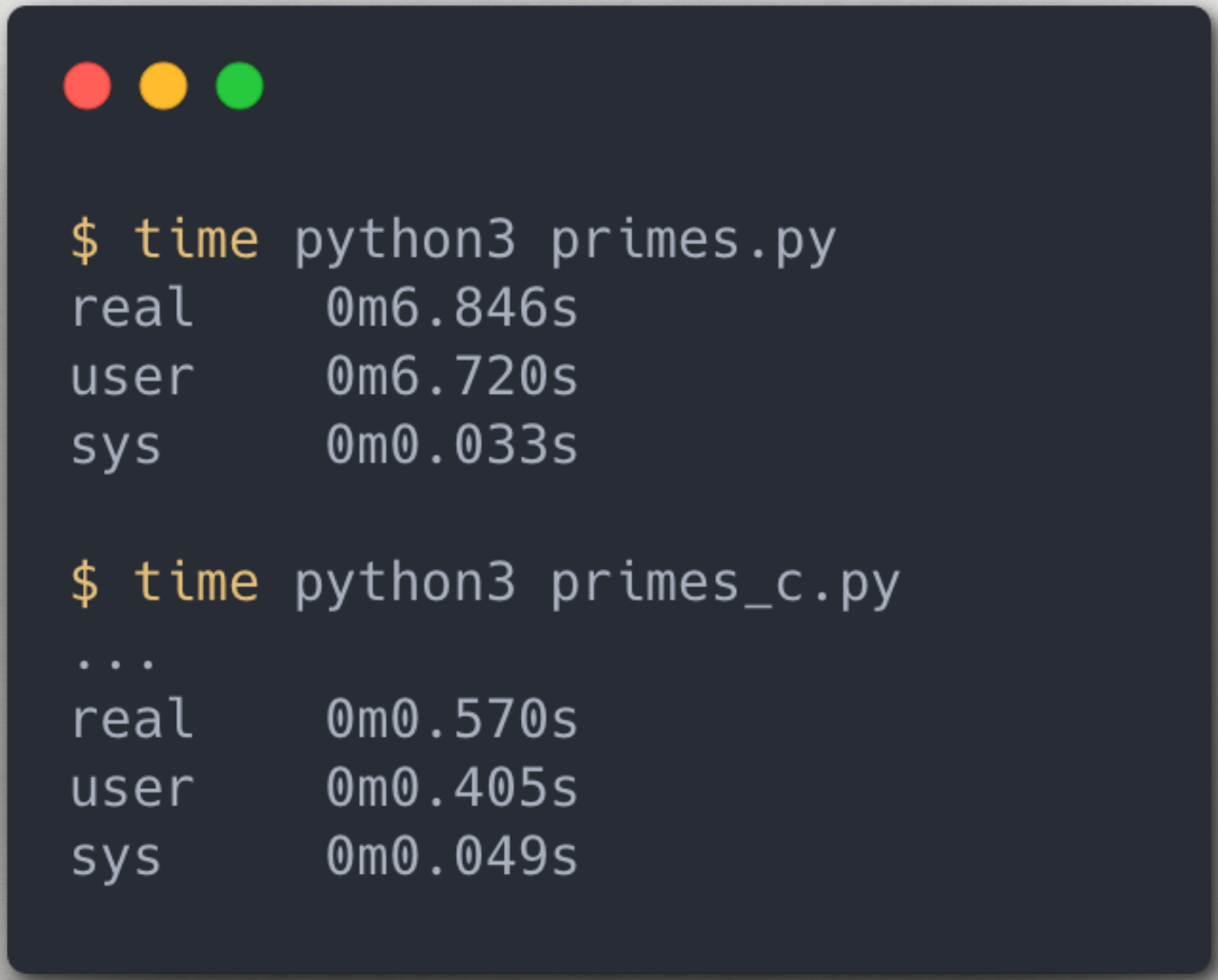
```
def check_if_prime(int number):
    cdef int i = 2

    while i < number:
        if number % i == 0:
            return False
        i += 1

    return True
```

regular code

```
def check_if_prime(number):

    for i in range(2, number):
        if number % i == 0:
            return False

    return True
```

TIME COMPARISON

REMEMBER TO RUN PRIMES_C.PY TWICE!



SUMMARY

1. Basic Python
   - Using venv
   - Third-party packages
   - Command Line Interface
   - Run external commands
2. Dealing with files
   - Open and close files
   - Deal with CSV files
   - Create graphics
   - Write PDFs, Word files and HTML
3. Efficient data treatment
   - Data types
   - Intro to Pandas
   - Cython

THANK YOU

THANK YOU

buelta.com

@jaimebuelta

Podcast wrongsideoflife.com