How can Python help your research

ResBaz 2020: Pick n Mix

Mike Laverick mike.laverick@auckland.ac.nz

Centre for eResearch @ UoA

Where can I learn more? (moved to front slides)

The internet is full of great resources to learn Python!
(way better than I could teach you) here are some links I found:

https://docs.python-guide.org/intro/learning/ (some great links to learning resources)

https://en.wikiversity.org/wiki/Python_Concepts (overview of way more Python concepts)

https://swcarpentry.github.io/python-novice-inflammation/ (great hands-on tutorials)

https://www.educba.com/python-programming-beginners-tutorial/

https://medium.com/fintechexplained/everything-about-python-from-beginner-to-advance-level-227d52ef32d2

https://realpython.com/jupyter-notebook-introduction/

Where can I learn more? (moved to front slides)

The internet is full of great resources to learn Python!
(way better than I could teach you) and some more others found:

https://www.youtube.com/watch?v=8DvywoWv6fI&list=PLY9xW0dssvfYuTAVS7eNoqhLdcsu291LI
https://pandas.pydata.org/pandas-docs/stable/getting_started/comparison/comparison_with_r.html
https://colab.research.google.com/
https://rstudio.com/solutions/r-and-python/
https://runestone.academy/runestone/books/published/thinkcspy/index.html

https://numpy.org/doc/stable/user/numpy-for-matlab-users.html

http://swcarpentry.github.io/python-novice-gapminder/

https://exercism.io/tracks/python

https://online-learning.harvard.edu/course/using-python-research

Example of Jupyter Notebooks / Google Colab

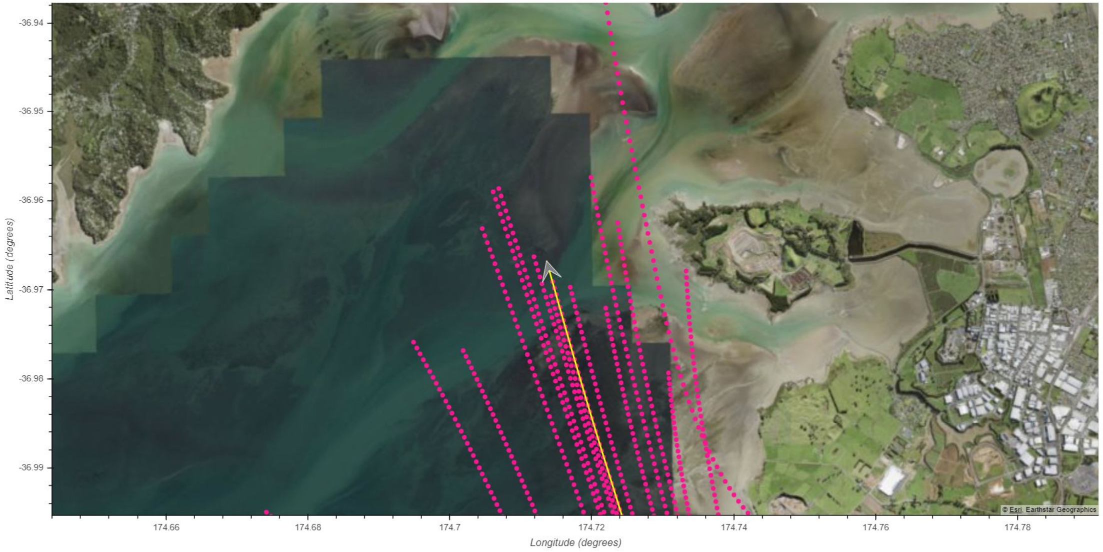
Here’s a hands-on Google Colab / Jupyter Notebook example for you to play with, check it out!

https://colab.research.google.com/drive/1eM58YLvUuUNG-ohNN25fnlcqew6jeD3z?usp=sharing

HELLO!

I am Mike Laverick

I am an eResearch Solution Specialist at
the Centre for eResearch, University of Auckland



1

What is “Python”?

It’s more than just a snake

Overview of the Python language

- Object-orientated language
  It’s great for representing real-world objects: i.e. things with attributes
- Interpreted language
  no need to compile the code every time you make a change
- Open-source language
  It’s free, well supported, and works across all major Operating Systems (windows, Macintosh, linux)

Overview of the Python language

- Extensive community libraries

  Many existing free codes that you can use as part of your own work

- Easy to learn

  One of the easiest programming languages to learn

- 2nd most popular programming language in the world!

  Having just taken #2 from Java in Nov 2020, breaking Java's 20 year streak

> “*Python is for the age where computers are cheap and ~~programmers~~ are expensive. Researchers*

Quick and easy to write useful code!

It’s a Python world

Python runs a substantial part of our everyday lives

Everyday companies using Python

Google

"Python where we can, C++ where we must."
Google co-founders
Larry Page & Sergey Brin

Spotify

80% of the Spotify app backend is written in Python: used to handle process logistics and machine-learning for song recommendations

Dropbox

Almost all of Dropbox runs using Python. They even employ the creator of Python, *Guido van Rossum*, to keep things optimised

Uber

"These first languages (Node.js & Python) still power most services running at Uber today."

Instagram

"Instagram currently features the world's largest deployment of the Django web framework, which is written entirely in Python."

Netflix

Netflix use Python to monitor their services and operations, as well as data science insights and visualisations into user behaviour

2

So what can Python do?

And how can it help with research?

What can Python do?

Handle data

Python can handle pretty much all file types: *text, CSV, binary, images* and many bespoke formats used in research (via comm. packages)

Complex & large data

Python is made for complex data structures, and many packages deal specifically with large data: *Pandas, hdf5, pickle, databases*

Data analysis

Python has a wealth of community packages dedicated to all kinds of analysis and data science: *numpy, scipy,* and *field-specific* ones

Visualisation

Python has a wide range of visualisation packages for 2D, 3D, interactive, & large-volume data plots *Matplotlib, Seaborn, Plotly, Bokeh*

Machine learning

One of the modern draws to Python are it's machine learning capabilities via: *NumPy, SciPy, Scikit-learn, PyTorch, TensorFlow*

Graphical User Interfaces

Python can also make powerful user interfaces for more hands-on programming and analysis

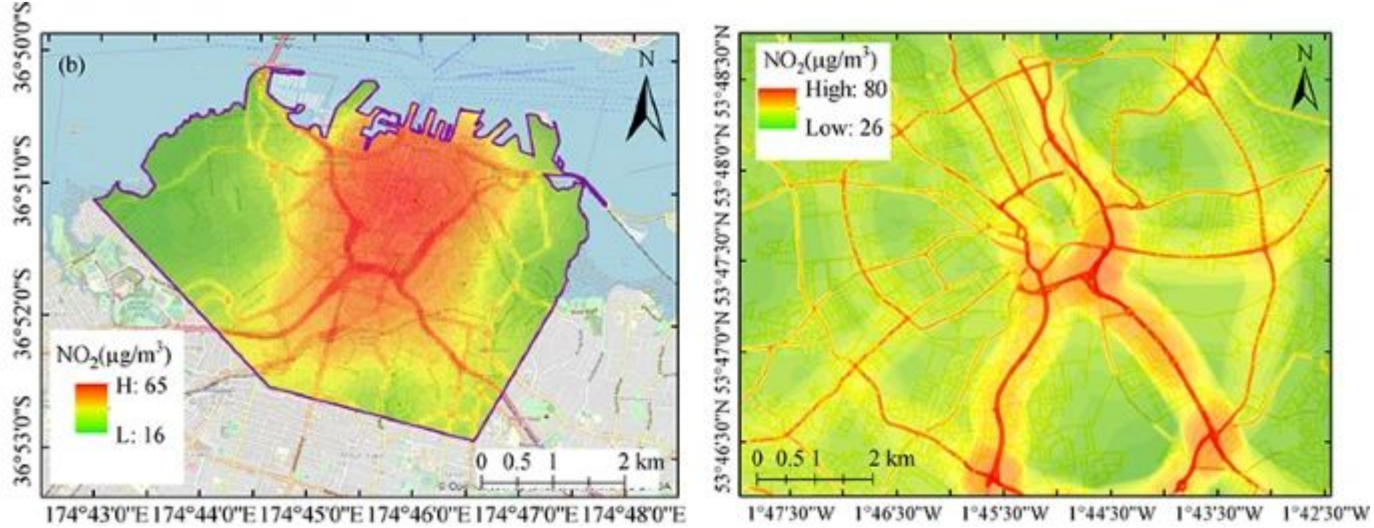
Python in research

Examples of Python in research

Some examples of Research at the UoA using Python include:

From 2020 alone!

Geospatial modelling of pollutants over cities


36°50'0"S
36°51'0"S
36°52'0"S
36°53'0"S
(b)
N
$NO_2(\mu g/m^3)$
H: 65
L: 16
0 0.5 1 2 km
174°43'0"E 174°44'0"E 174°45'0"E 174°46'0"E 174°47'0"E 174°48'0"E
53°48'30"N 53°48'0"N 53°47'30"N 53°47'0"N 53°46'30"N
$NO_2(\mu g/m^3)$
High: 80
Low: 26
N
0 0.5 1 2 km
1°47'30"W 1°46'30"W 1°45'30"W 1°44'30"W 1°43'30"W 1°42'30"W

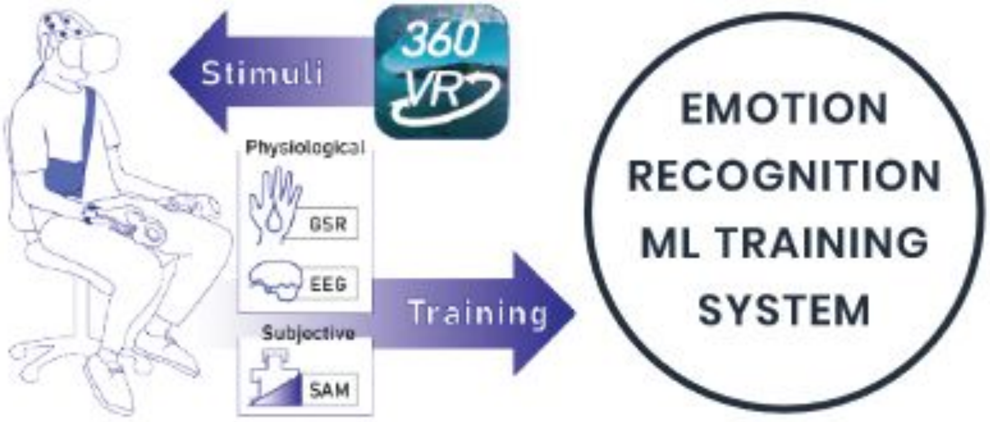
Community software package

Ma, X., Longley, I., Salmond, J. *et al.* PyLUR: Efficient software for land use regression modeling the spatial distribution of air pollutants using GDAL/OGR library in Python. *Front. Environ. Sci. Eng.* 14, 44 (2020). https://doi.org/10.1007/s11783-020-1221-5

Some examples of Research at the UoA using Python include:

From 2020 alone!

Machine learning (ML) to model emotional responses to Virtual Reality



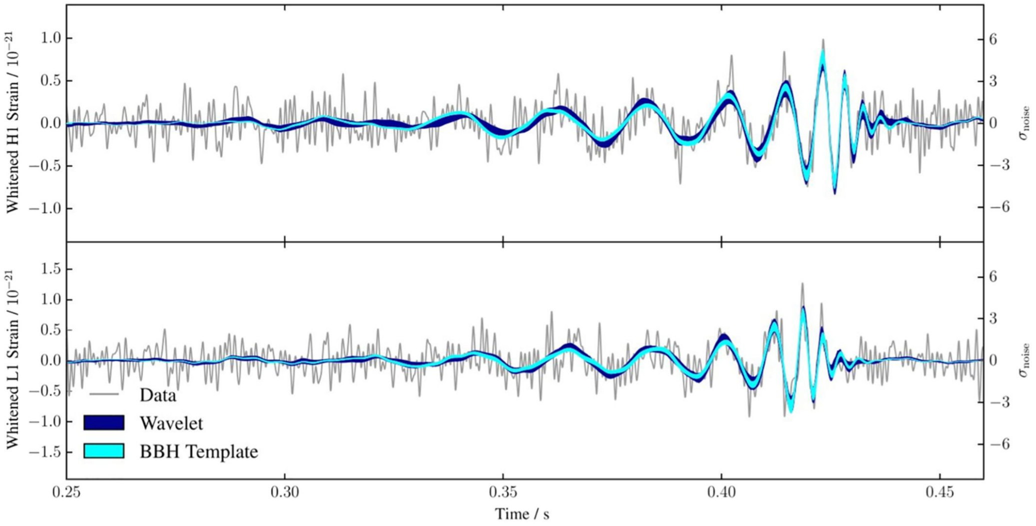
Training ML algorithms and data handling

Kunal Gupta, Jovana Lazarevic, Yun Suen Pai, and Mark Billinghurst. 2020. AffectivelyVR: Towards VR Personalized Emotion Recognition. In 26th ACM Symposium on Virtual Reality Software and Technology (VRST '20). Association for Computing Machinery, New York, NY, USA, Article 36, 1–3. DOI:https://doi.org/10.1145/3385956.3422122

Some examples of Research at the UoA using Python include:

From 2020 alone!

Parameter estimation of gravitational wave signals



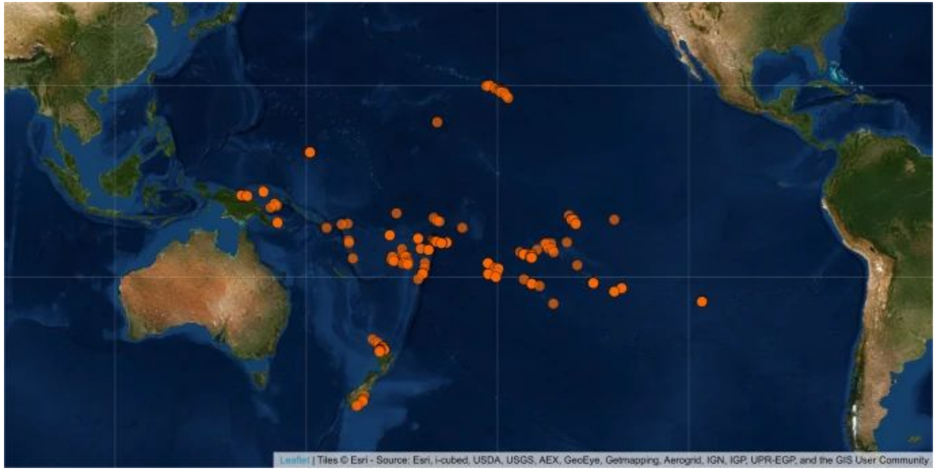
Markov-Chain Monte Carlo / Bayesian statistics

Meyer, R, Edwards, MC, Maturana-Russel, P, Christensen, N. Computational techniques for parameter estimation of gravitational wave signals. WIREs Comput Stat. 2020;e1532. https://doi.org/10.1002/wics.1532

Some examples of Research at the UoA using Python include:

From 2020 alone!

Open-access archaeological sample database



Leaflet | Tiles © Esri - Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

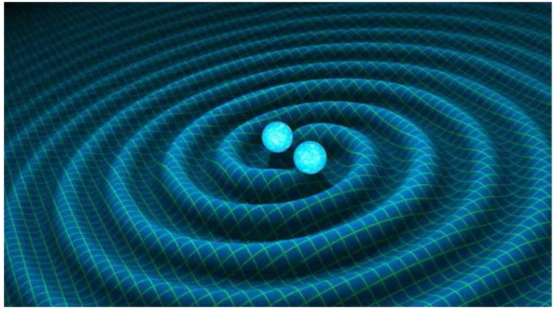
Creating interactive databases and visualisations

Hermann, A., Forkel, R., McAlister, A. et al. Pofatu, a curated and open-access database for geochemical sourcing of archaeological materials. Sci Data 7, 141 (2020). https://doi.org/10.1038/s41597-020-0485-8

Some examples of Research at the UoA using Python include:

From 2020 alone!

Creating convenient “wrappers” to run codes



hoki.

Running non-python codes...
within Python!

Stevance, H., Eldridge, J., and Stanway, E., “Hoki: Making BPASS accessible through Python”, The Journal of Open Source Software, vol. 5, no. 45, 2020. doi:10.21105/joss.01987.

It’s versatile!

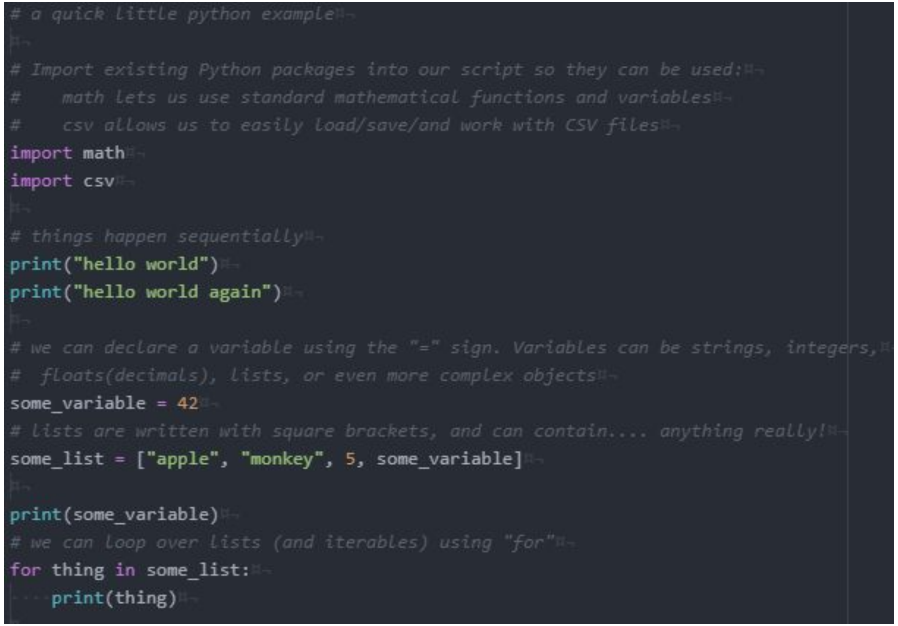
An easy to write, quick to develop, and multi-functional language

3

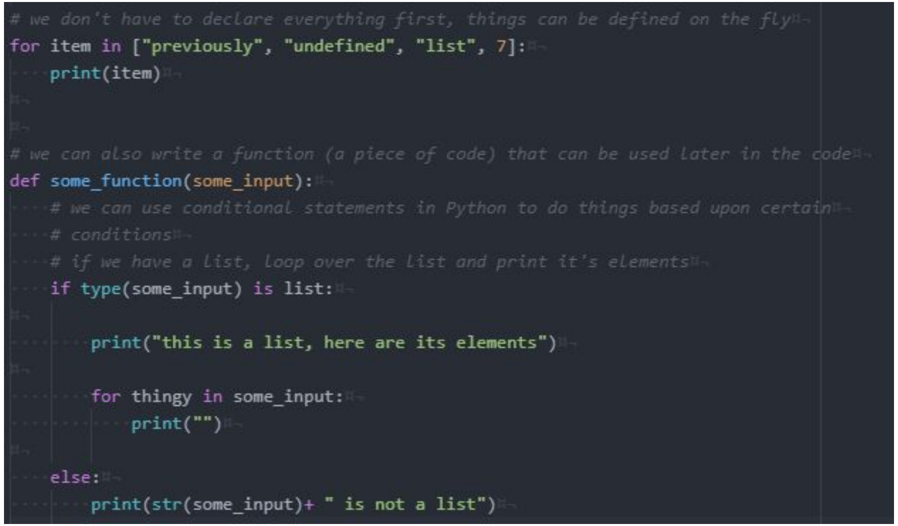
Key concepts in Python

Practical things to know about Python

Key concepts: this is a Python code file (text file)



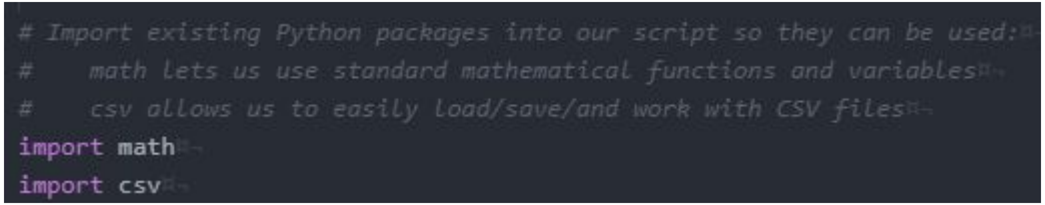
```python
# a quick little python example

# Import existing Python packages into our script so they can be used:
#    math lets us use standard mathematical functions and variables
#    csv allows us to easily load/save/and work with CSV files
import math
import csv

# things happen sequentially
print("hello world")
print("hello world again")

# we can declare a variable using the "=" sign. Variables can be strings, integers,
#  floats(decimals), lists, or even more complex objects
some_variable = 42
# lists are written with square brackets, and can contain.... anything really!
some_list = ["apple", "monkey", 5, some_variable]

print(some_variable)
# we can loop over lists (and iterables) using "for"
for thing in some_list:
    print(thing)
```



```python
# we don't have to declare everything first, things can be defined on the fly
for item in ["previously", "undefined", "list", 7]:
    print(item)


# we can also write a function (a piece of code) that can be used later in the code
def some_function(some_input):
    # we can use conditional statements in Python to do things based upon certain
    # conditions
    # if we have a list, loop over the list and print it's elements
    if type(some_input) is list:

        print("this is a list, here are its elements")

        for thingy in some_input:
            print("")

    else:
        print(str(some_input)+ " is not a list")
```

1 Key concepts: importing modules/packages

You can “import” existing python code into your own code



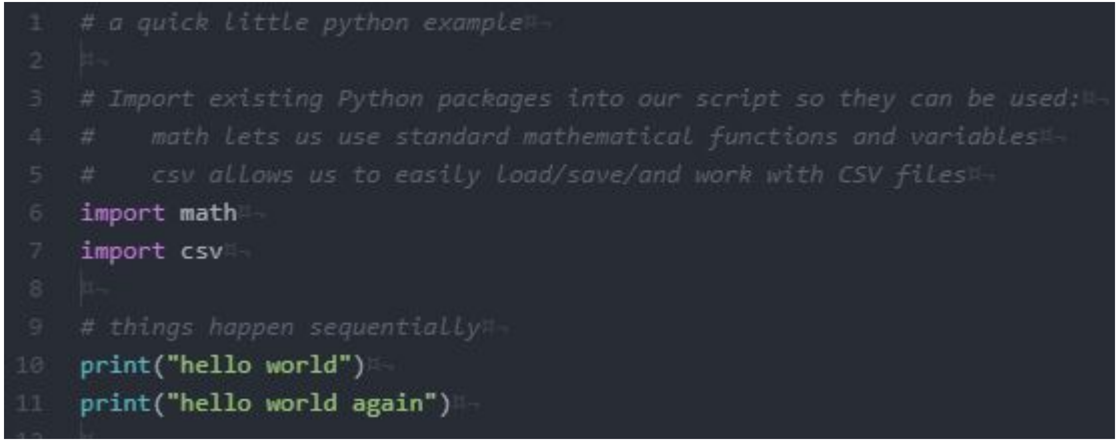
```
# Import existing Python packages into our script so they can be used:
#    math lets us use standard mathematical functions and variables
#    csv allows us to easily load/save/and work with CSV files
import math
import csv
```

Python comes with many “standard libraries” for commonly-used codes (pre-made packages)

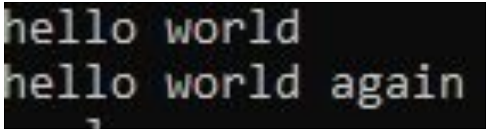
If in doubt, check if someone has already done it!

2

Key concepts: things happen sequentially



```python
# a quick little python example

# Import existing Python packages into our script so they can be used:
#     math lets us use standard mathematical functions and variables
#     csv allows us to easily load/save/and work with CSV files
import math
import csv

# things happen sequentially
print("hello world")
print("hello world again")
```

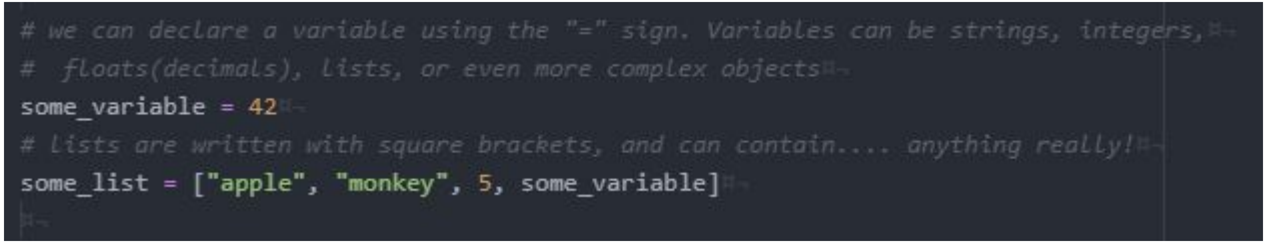


```
hello world
hello world again
```

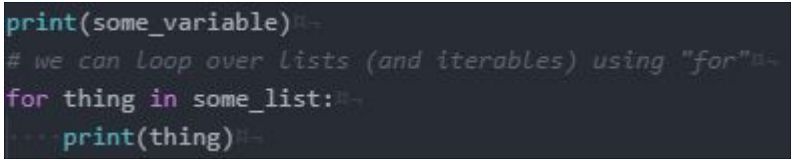
Print() displays things within the brackets

3 Key concepts: defining variables/objects

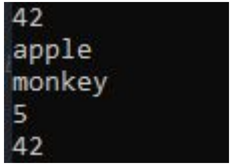
You can define variables, lists (and more) and do things with them



```python
# we can declare a variable using the "=" sign. Variables can be strings, integers,
#  floats(decimals), lists, or even more complex objects
some_variable = 42
# lists are written with square brackets, and can contain.... anything really!
some_list = ["apple", "monkey", 5, some_variable]
```



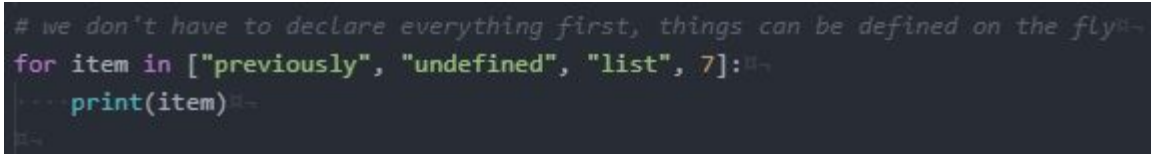
```python
print(some_variable)
# we can loop over lists (and iterables) using "for"
for thing in some_list:
    print(thing)
```



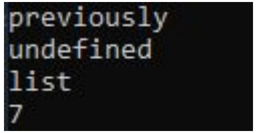
```
42
apple
monkey
5
42
```

3 Key concepts: defining variables/objects

You can define variables, lists (and more) and do things with them



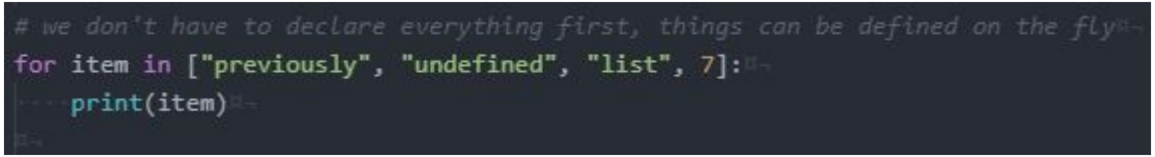
```python
# we don't have to declare everything first, things can be defined on the fly
for item in ["previously", "undefined", "list", 7]:
    print(item)
```



```
previously
undefined
list
7
```

4 Key concepts: whitespace denotes “blocks”

Unlike many other languages (C, Java, etc) Python cares about whitespace



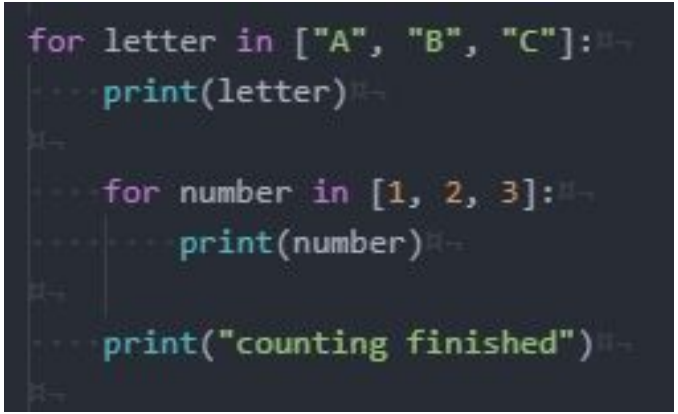
```python
# we don't have to declare everything first, things can be defined on the fly
for item in ["previously", "undefined", "list", 7]:
    print(item)
```

Python uses indent (whitespace) size to determine which sections of code are grouped together in a “block”, instead of placing brackets around the sections

4

Key concepts: whitespace denotes “blocks”

We already saw a loop examp

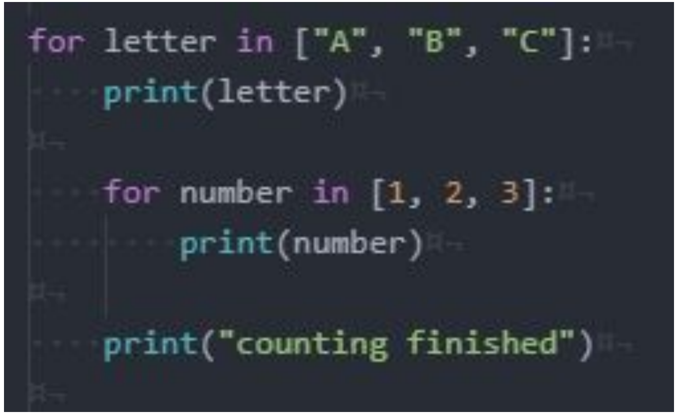


```
for letter in ["A", "B", "C"]:
    print(letter)

    for number in [1, 2, 3]:
        print(number)

    print("counting finished")

```

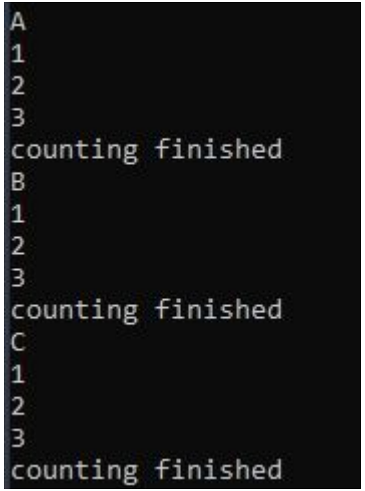
- Chapter one
  - Subchapter one
    - Paragraph 1
    - Paragraph 2
  - Subchapter two
    - Paragraph 1
- Chapter two

4 Key concepts: whitespace denotes “blocks”

We already saw a loop example; now lets increase the complexity



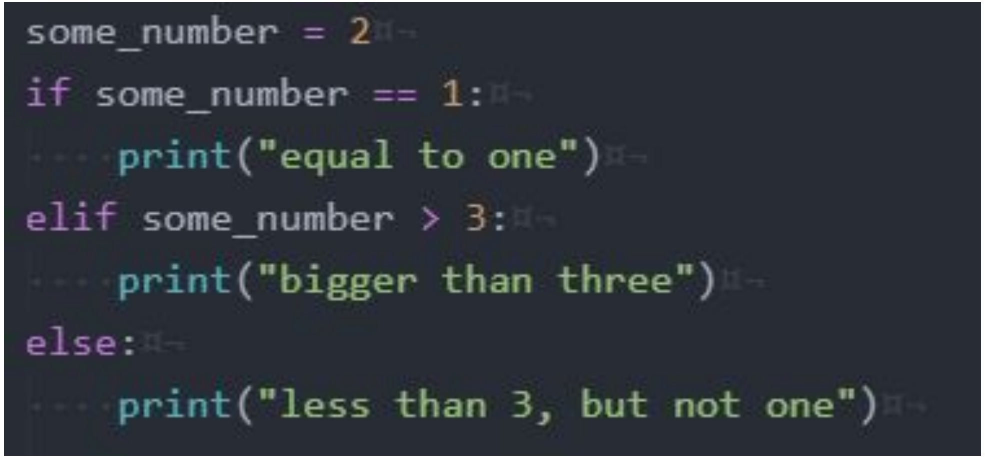
```
for letter in ["A", "B", "C"]:
    print(letter)

    for number in [1, 2, 3]:
        print(number)

    print("counting finished")
```



```
A
1
2
3
counting finished
B
1
2
3
counting finished
C
1
2
3
counting finished
```

5 Key concepts: conditionals

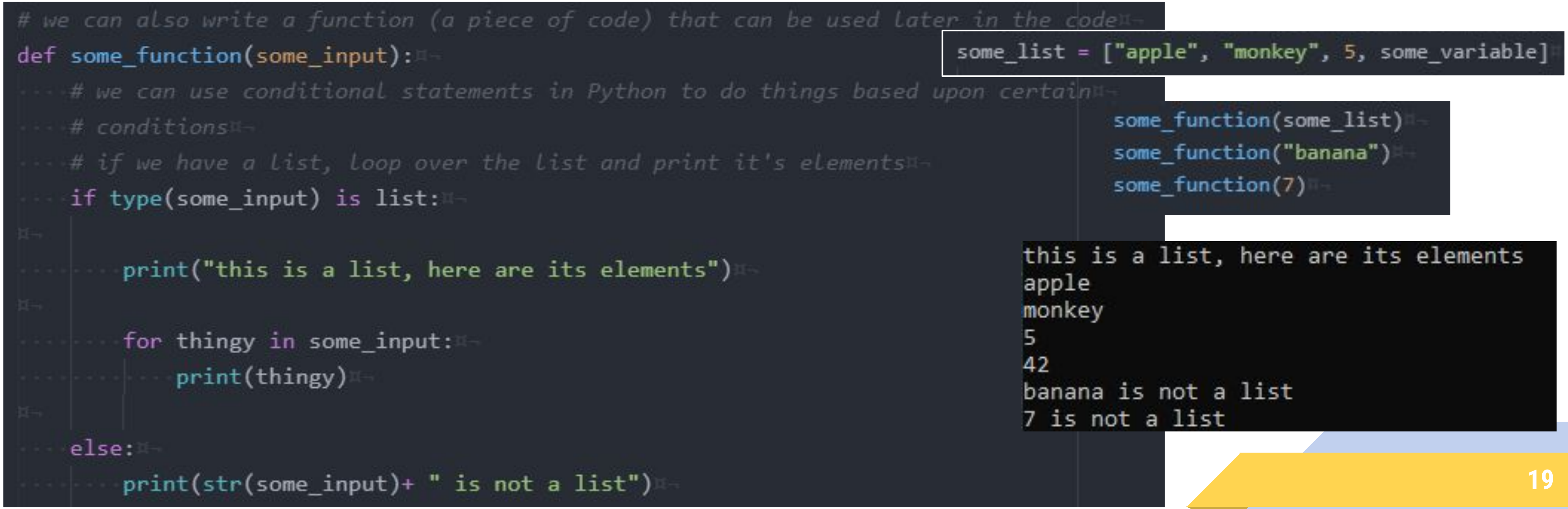
Not unique to Python, but a powerful & important concept: logic



```python
some_number = 2
if some_number == 1:
    print("equal to one")
elif some_number > 3:
    print("bigger than three")
else:
    print("less than 3, but not one")
```

6 Key concepts: functions

You can create a function that can be used later in your code when called upon (def = define)



```python
# we can also write a function (a piece of code) that can be used later in the code
def some_function(some_input):
    # we can use conditional statements in Python to do things based upon certain
    # conditions
    # if we have a list, loop over the list and print it's elements
    if type(some_input) is list:

        print("this is a list, here are its elements")

        for thingy in some_input:
            print(thingy)

    else:
        print(str(some_input)+ " is not a list")
```

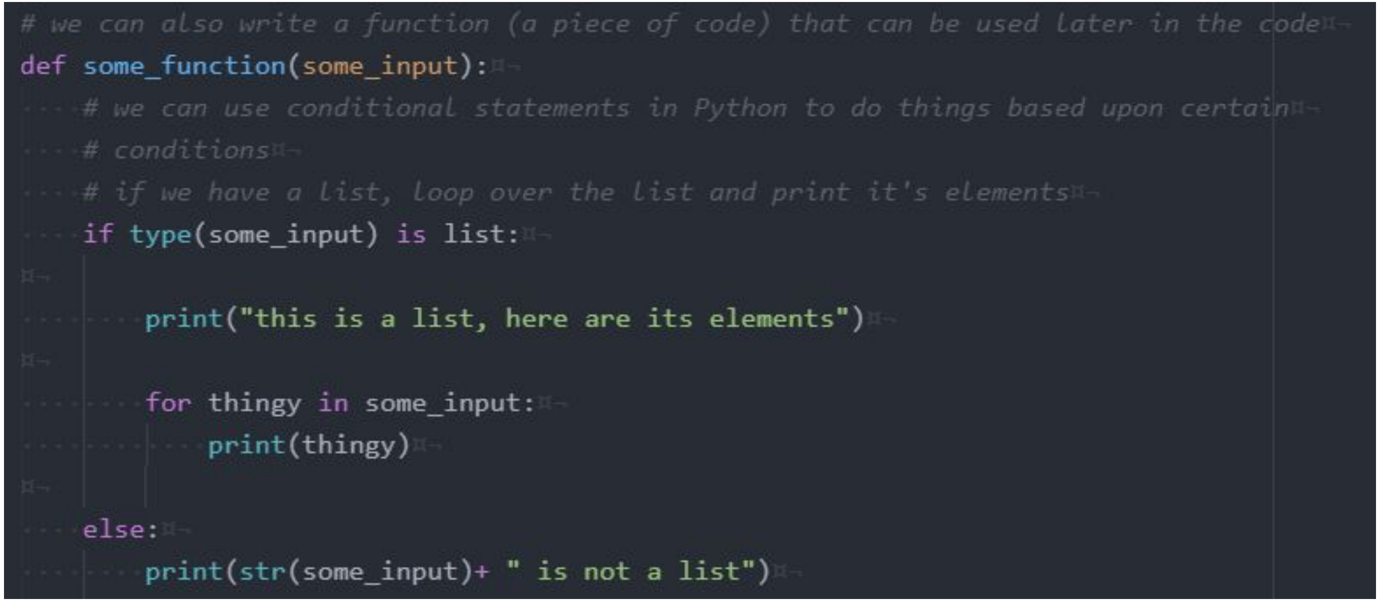
```python
some_list = ["apple", "monkey", 5, some_variable]
```

```python
some_function(some_list)
some_function("banana")
some_function(7)
```

```
this is a list, here are its elements
apple
monkey
5
42
banana is not a list
7 is not a list
```

6 Key concepts: functions

You can create a function that can be used later in your code when called upon (def = define)



```python
# we can also write a function (a piece of code) that can be used later in the code
def some_function(some_input):
    # we can use conditional statements in Python to do things based upon certain
    # conditions
    # if we have a list, loop over the list and print it's elements
    if type(some_input) is list:

        print("this is a list, here are its elements")

        for thingy in some_input:
            print(thingy)

    else:
        print(str(some_input)+ " is not a list")
```

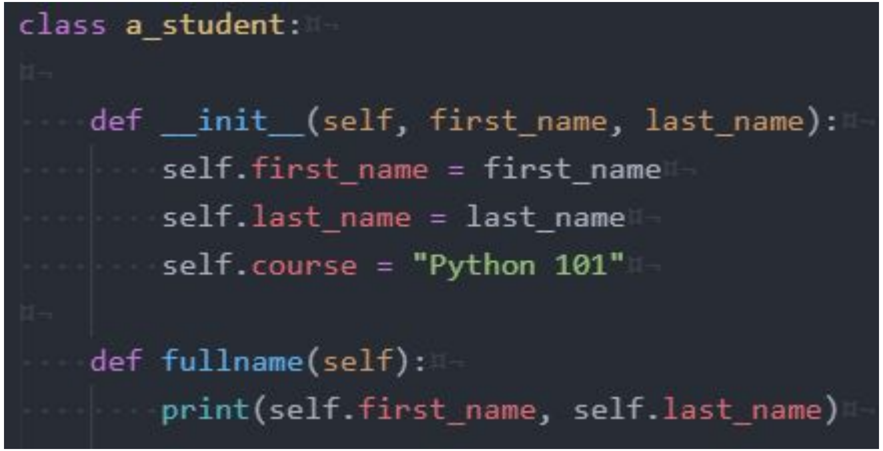
(This is pretty much how modules & packages work!)

Beyond the basics: Python objects called “Classes”

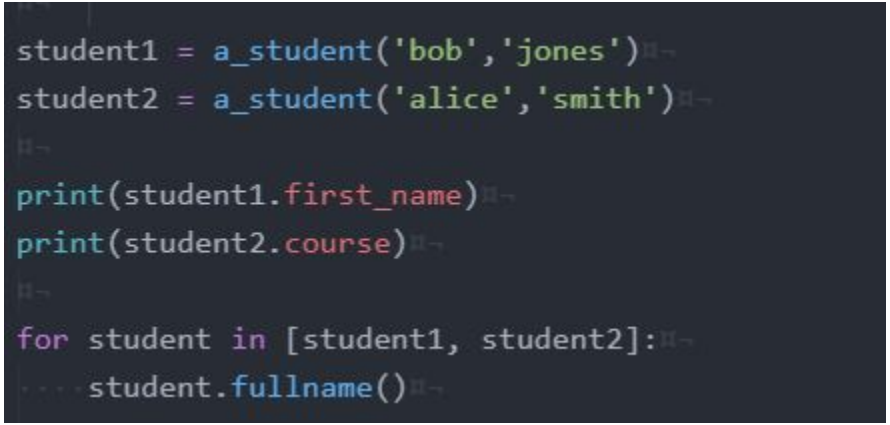
- Classes are powerful objects that can be used to group variables/functions/attributes together

- They can be used as a “template” to describe properties of an object. Instances of the object can have some common properties, and also individual properties

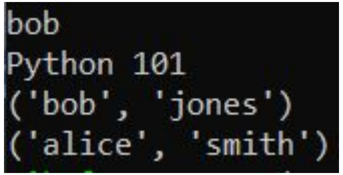
Beyond the basics: Python objects called “Classes”



```
class a_student:

    def __init__(self, first_name, last_name):
        self.first_name = first_name
        self.last_name = last_name
        self.course = "Python 101"

    def fullname(self):
        print(self.first_name, self.last_name)
```



```
student1 = a_student('bob','jones')
student2 = a_student('alice','smith')

print(student1.first_name)
print(student2.course)

for student in [student1, student2]:
    student.fullname()
```



```
bob
Python 101
('bob', 'jones')
('alice', 'smith')
```

They’re powerful, but you may not need to use them (I use them often now, but I never used one during my PhD!)

That’s enough!

Those were some of the basic, but important, concepts!

4

How to get started with python

And learn to write some code

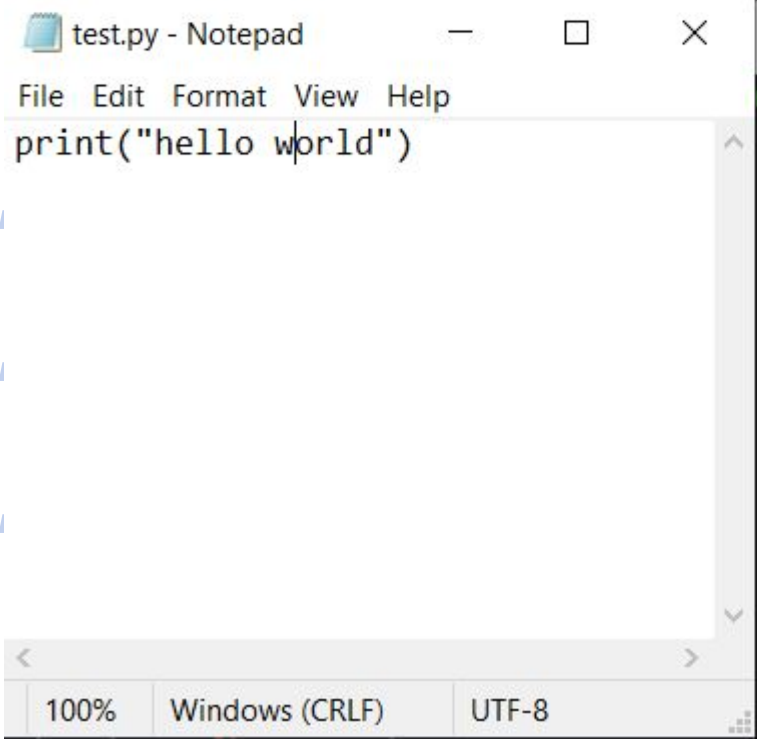
What do I need?

There are a few ways to write and run a Python file:

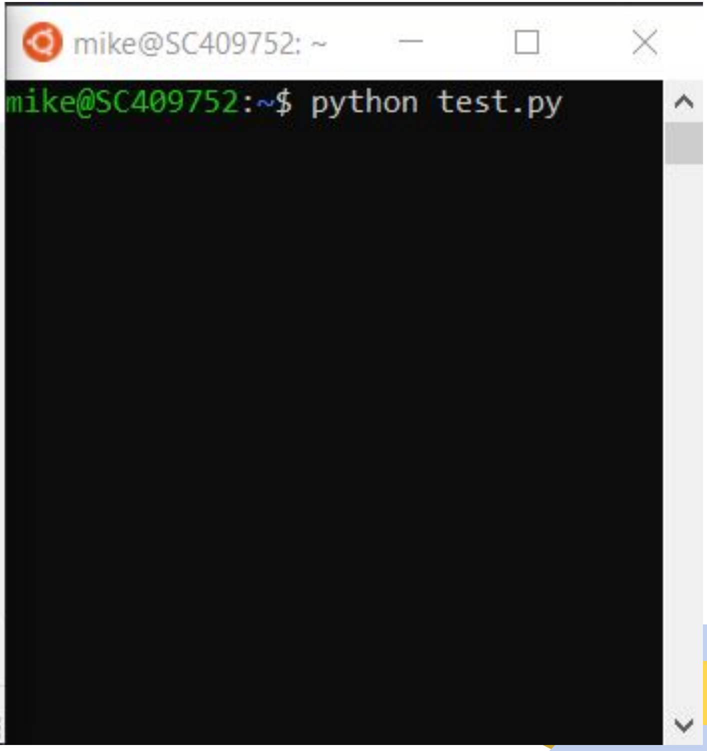
- A plain text editor + command line
- IDE (Integrated Development Environment)
- Python Shell itself (via command line)

What do I need?

There

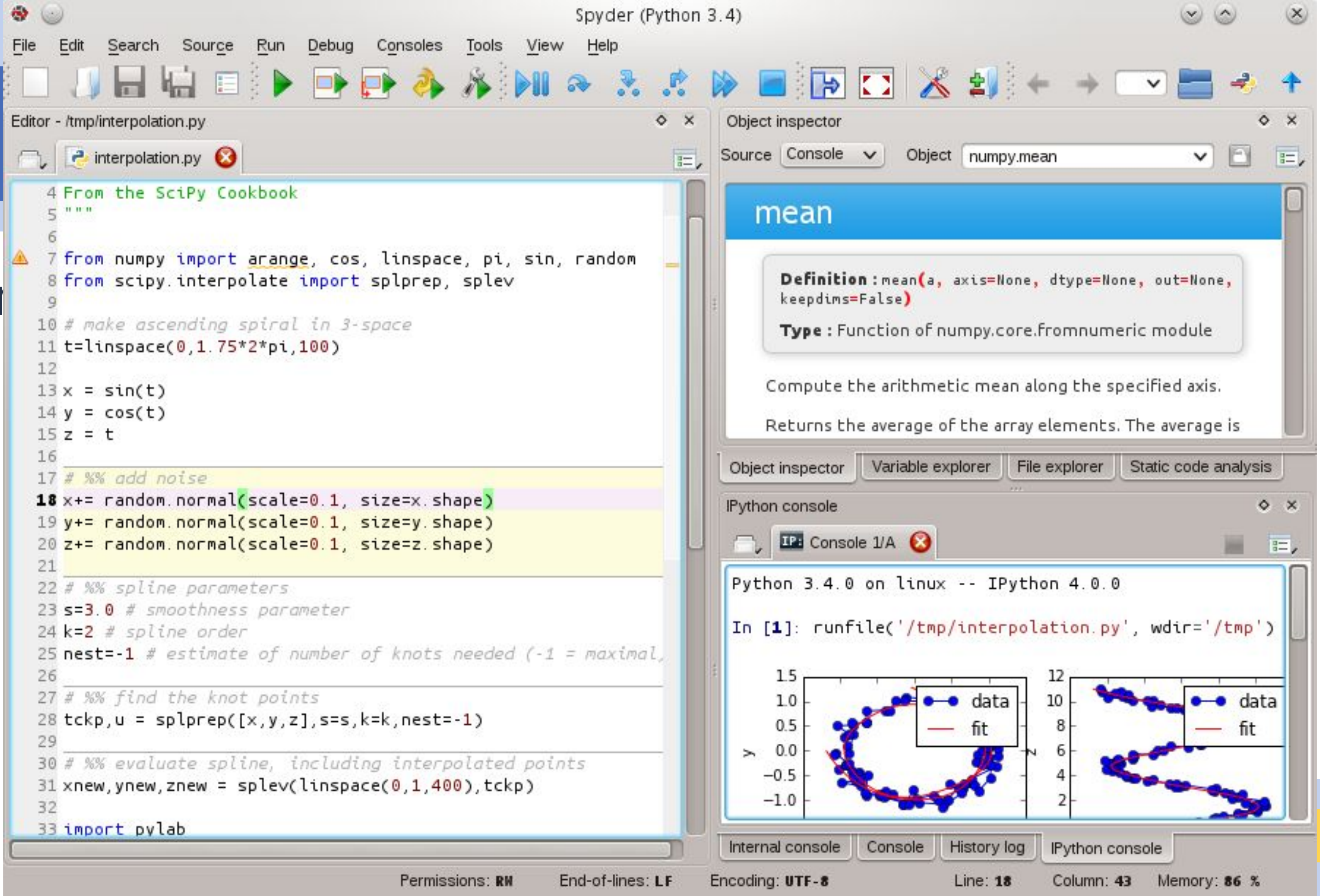


```
print("hello world")
```



```
mike@SC409752:~$ python test.py
```

Ther



```
From the SciPy Cookbook
"""

from numpy import arange, cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# make ascending spiral in 3-space
t=linspace(0,1.75*2*pi,100)

x = sin(t)
y = cos(t)
z = t

# %% add noise
x+= random.normal(scale=0.1, size=x.shape)
y+= random.normal(scale=0.1, size=y.shape)
z+= random.normal(scale=0.1, size=z.shape)

# %% spline parameters
s=3.0 # smoothness parameter
k=2 # spline order
nest=-1 # estimate of number of knots needed (-1 = maximal,

# %% find the knot points
tckp,u = splprep([x,y,z],s=s,k=k,nest=-1)

# %% evaluate spline, including interpolated points
xnew,ynew,znew = splev(linspace(0,1,400),tckp)

import pylab
```

mean

Definition : `mean(a, axis=None, dtype=None, out=None, keepdims=False)`

Type : Function of numpy.core.fromnumeric module

Compute the arithmetic mean along the specified axis.

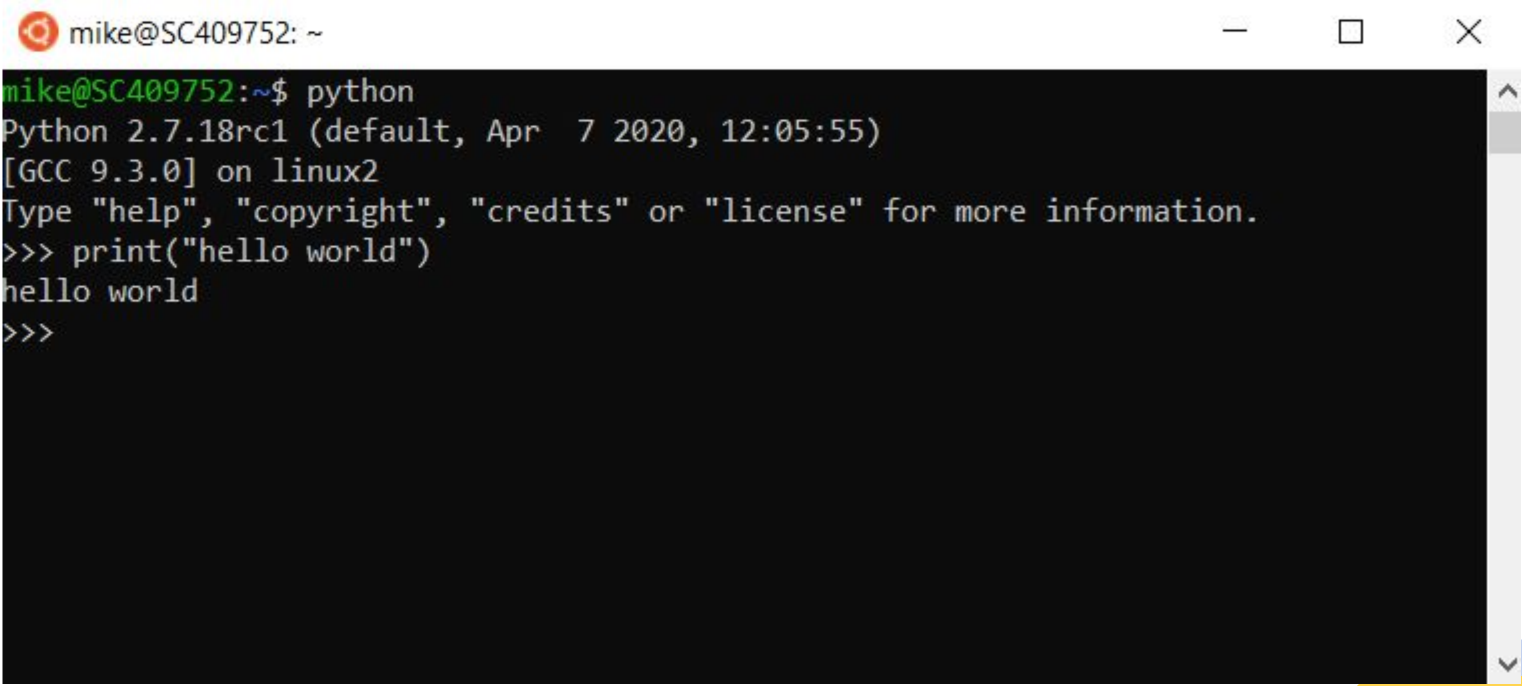
Returns the average of the array elements. The average is

```
Python 3.4.0 on linux -- IPython 4.0.0

In [1]: runfile('/tmp/interpolation.py', wdir='/tmp')
```

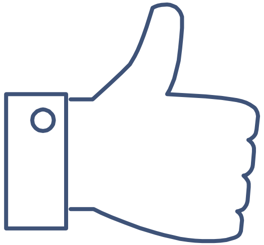
Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 18 Column: 43 Memory: 86 %

What do I need?

The



```
mike@SC409752:~$ python
Python 2.7.18rc1 (default, Apr  7 2020, 12:05:55)
[GCC 9.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello world")
hello world
>>>
```



THANKS!

Time to answer any questions you have!

You can find me at

mike.laverick@auckland.ac.nz

Or at HackyHour!

Original presentation template design by SlidesCarnival