UNIVERSITY OF
OXFORD

TB5 – Introduction To Python Programming

Dr Irfan Alibay

Demonstrators: Bjarne Feddersen

Course Aims and Objectives

- Provide a simple first impression of the Python programming language
  - Fully learning Python is a multi-months to years task!
  - Primer for further studies in Python and other programming languages

- Introduce the basic concepts of Python
  - Base types, using and manipulating variables, loops, etc…

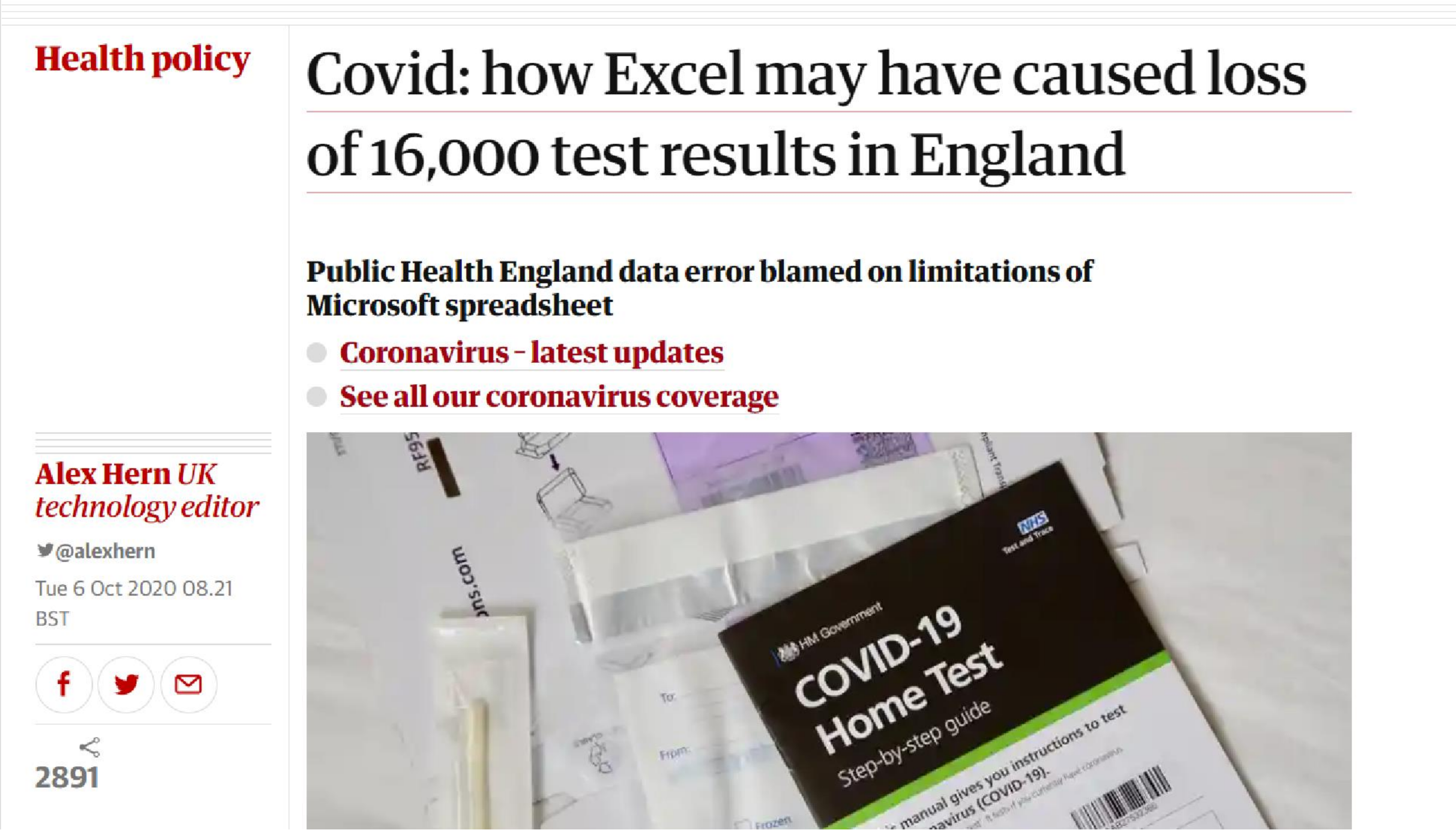
Why does this tutorial exist?

- Software is an increasingly important in science
  - Data analysis
  - Machine learning
  - Simulations / modelling

- Mixes well with the concepts shown in the Molecular Dynamics tutorial later in the week

- Fun concept which can be used to handle (non-scientific) everyday tasks!

Why (Python) programming?



Health policy

Covid: how Excel may have caused loss of 16,000 test results in England

Public Health England data error blamed on limitations of Microsoft spreadsheet

- Coronavirus - latest updates
- See all our coronavirus coverage

Alex Hern *UK technology editor*

@alexhern

Tue 6 Oct 2020 08.21 BST

2891

The Guardian, 6th Oct 2020

Why (Python) programming?

- Automation of tasks
  - e.g. automated chemistry, analysis (e.g. ITC), model building
- Reproducible and transparent data processing
  - “Big data” and “machine learning” are a reality of modern science
- Creation and use of many widely used scientific tools
  - Pymol, Chimera, OpenMM, Rosetta, etc…

The Python programming language

- 31 years old but ever-evolving
  - Official docs: https://docs.python.org/3/
  - Many built-in libraries, methods, etc…
  - Plenty of tools & libraries which rely / use Python
    - Molecular Dynamics: OpenMM, GSD, etc…
    - Machine learning; pytorch, tensorflow, etc…
    - SymPy, AstroPy, SunPy, QuantEcon, etc…

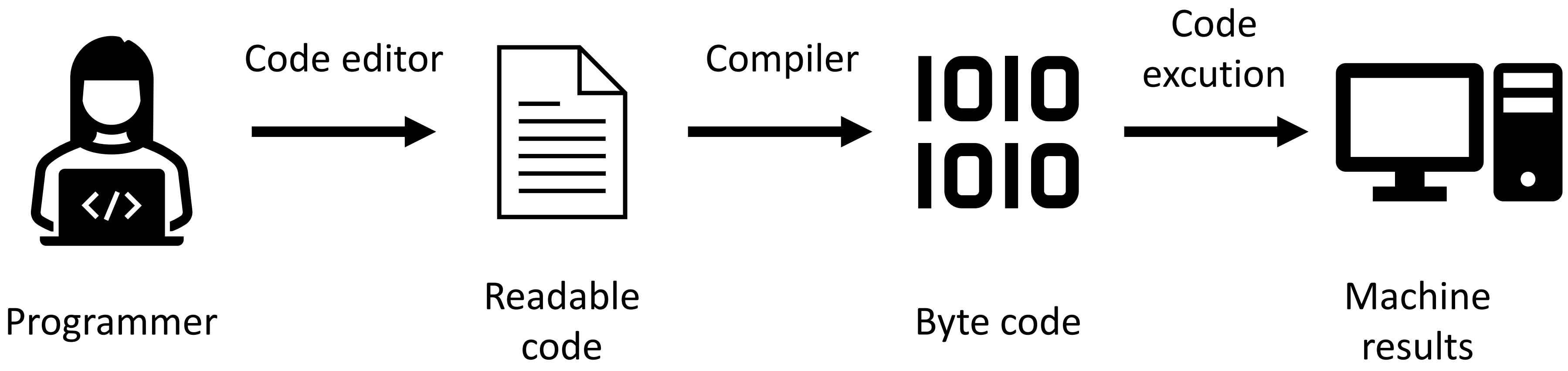
- Code executed via the `python` interpreter
  - Pass raw code to `python` and it just gets executed
  - Creation of “machine instructions” done on the fly (~ish)
    - Python is an interpreted language

How we use Python for research

- Creating frameworks for executing and handling molecular simulations
  - Take input data from wet lab experiments and use software to generate models
  - Write Python code to generate new data from these models and extract relevant features
  - Better context after Molecular Dynamics tutorial

- Involved in the development of several code libraries
  - Workflow for estimating free energy differences in ligand binding
    - https://openfree.energy/
  - Library for the handling and analysis of molecular simulation data
    - https://www.mdanalysis.org/

How programming languages work



Programmer

Code editor

Readable code

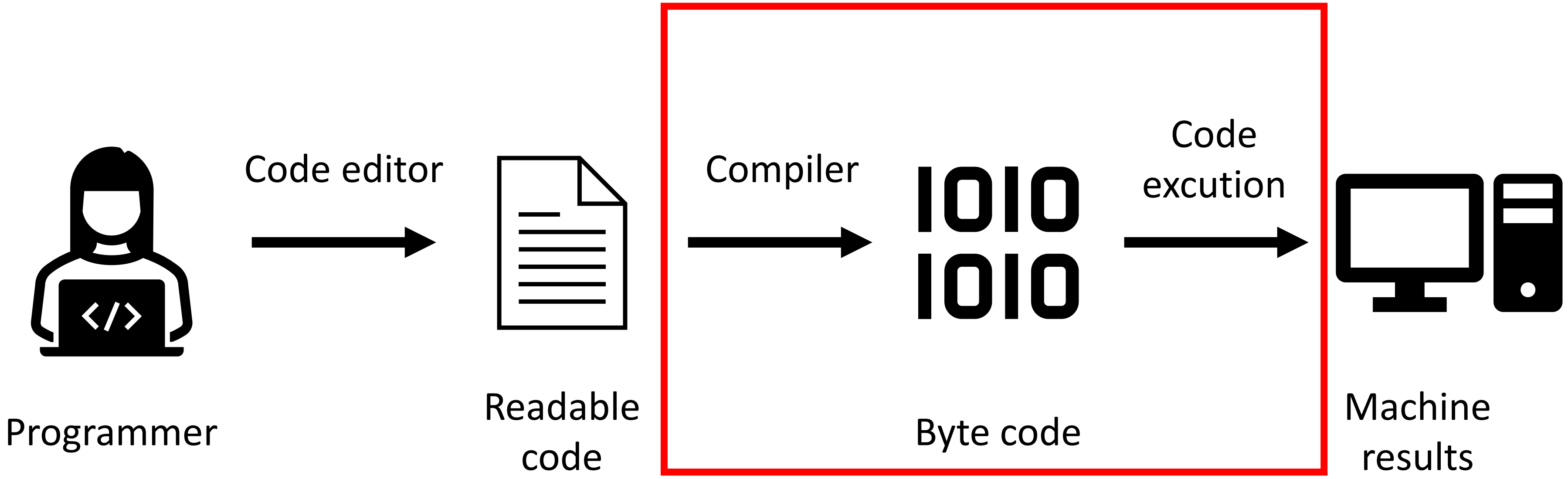
Compiler

Byte code

Code excution

Machine results

How programming languages work



- “interpreted” languages (like Python) do both in one go
    - “Python interpreter” deals with this
    - unlike “compiled” languages which do them separately
    - speed vs flexibility

Using Python

- Script-based execution
  - Write a file, let’s say with `print(“Hello World”)`
  - Call `python` to execute file `python file.py`, get your output (i.e. “Hello World”)

- Jupyter notebooks (today’s main tool)
  - Interactive way to execute snippets of python on-the-fly
  - Useful for visualization and prototyping
  - Not intended for production work!
  - *Note: slowly getting replaced by Jupyter lab*

A few Python gotchas (there are many)

- Formatting is very important
  - Spaces/tabs in code have a defined use!
  - Type of quote you use is important (i.e. “ vs ‘ vs " are all different)
  - Different types of brackets do different things
    - “()” is used for accessing methods
    - “[]” is used for accessing data structures (lists, dictionaries, sets, etc...)

- Others
  - Reasonably quick release schedule (yearly)
  - Complex library dependencies
    - Using a package manager like `conda` is a must!

Alternatives to python

UNIVERSITY OF OXFORD

- Use the right tool for the right use case
  - Transferable skillset

- High-level / interpreted languages
  - Matlab/Octave and R

- Low-level / high performance
  - Fortran, C/C++ (w/ extra libraries; MPI, OpenMP, CUDA…)

Launching the notebooks

- Instructions provided on Canvas
  1. Start the Linux VM
     - Password is: **BioComp**
  2. Open a terminal
  3. Activate the “tb5-env” environment
     - “conda activate tb5-env”
  4. Start Jupyter notebook
     - “jupyter notebook”

Using Jupyter demo

~ Quick live demo of using a notebook ~

Access to notebooks beyond today

- Notebook is open source (CC-BY-4.0)
  - Hosted here: https://github.com/bigginlab/TB5-IntroductionToPython

- You can run the notebooks from home using either Google Colab or Mybinder
  - Instructions provided in Canvas and notebook, or just click on one of the badges on Github!

- This is a new notebook!
  - Please let us know your thoughts, we'd love to improve it over time
    - In-person, email, or over Github as an issue

Colab demo

~ Quick live demo of using colab for the notebook ~

Exploring Python beyond today

- Python website & documentation
  - https://www.python.org/
  - https://docs.python.org/3/

- Online courses
  - Computational Biochemistry course: https://github.com/bigginlab/OxCompBio/tree/master/tutorials/Python
  - Carpentries Python workshop: https://swcarpentry.github.io/python-novice-inflammation/

- Look at other people's code (on Github, Gitlab, etc..)!
  - ~ Demo is feasible ~