Introduction to Programming

COMP0015

Admin & Introduction

Module Instructors

- Lecturers:
  - Dr Aquinas Hobor (a.hobor@ucl.ac.uk)
  - Dr Quang Loc Le (loc.le@ucl.ac.uk)
  - Dr Sobhan Y. Tehrani (sobhan.tehrani@ucl.ac.uk)
- Teaching Assistants for the labs
  - Hadrien Renaud (hadrien.renaud.22@ucl.ac.uk)
  - Mar Munoz (mar.munoz.22@ucl.ac.uk)
- Contact details and office hours on the Moodle page of the module

Schedule

UCL

	Monday 15:00 -17:00	Thursday 15:00 - 17:00	Fridays 13:00-15:00	Fridays 15:00-17:00
1	Introduction/Lab	Introduction/Lab	Introduction/Lab	Introduction/Lab
2	Pratical Session	Pratical Session	Pratical Session	Pratical Session
3	Pratical Session	Pratical Session	Pratical Session	Pratical Session
4	Pratical Session	Pratical Session	Pratical Session	Pratical Session
5	Pratical Session	Pratical Session	Pratical Session	Pratical Session
	Reading Week			
6	N/A	N/A	Exam	
7	Pratical Session	Pratical Session	Pratical Session	Pratical Session
8	Pratical Session	Pratical Session	Pratical Session	Pratical Session
9	Pratical Session	Pratical Session	Pratical Session	Pratical Session
10	Pratical Session	Pratical Session	Pratical Session	Pratical Session

CW

Deadline

	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
MON							computer practical COMP0015-A5U-T1 Introduction to Programming LE, Quang Loc (Dr), HOBOR, Aquinas (Dr), YASSIP... **Malet Place Engineering Building 1.21, Malet Pl...** 6-10, 13-16 GRP1			
TUE										
WED										
THU							computer practical COMP0015-A5U-T1 Introduction to Programming LE, Quang Loc (Dr), HOBOR, Aquinas (Dr), YASSIP... **Malet Place Engineering Building 1.21, Malet Pl...** 6-10, 13-16 GRP2			
FRI					computer practical COMP0015-A5U-T1 Introduction to Programming LE, Quang Loc (Dr), HOBOR, Aquinas (Dr), YASSIP... **Malet Place Engineering Building 1.21, Malet Pl...** 6-10, 13-16 GRP4		computer practical COMP0015-A5U-T1 Introduction to Programming LE, Quang Loc (Dr), HOBOR, Aquinas (Dr), YASSIP... **Malet Place Engineering Building 1.21, Malet Pl...** 6-10, 13-16 GRP3			

Support

UCL

- Lab Structure: 4 groups and 4 sessions
- Structure:
  - Watch the video lecture in advance
  - Students work through the content during the labs (be prepared!)
  - Attend the allocated session (group 1-4)
  - Support ask as needed

Office hours

UCL

- Monday 13:00-14:00 (online) - Dr Hobor
- Thursday 13:00-14:00 (online) - Dr Le
- Friday 11:00-12:00 (in person) - Dr Tehrani

Assessment

UCL

- 60% Coursework
  - Individual coursework (deadline: end of last week)
- 40% Exam
  - Written examination (after the reading week)

UCL Plagiarism Guidelines

UCL

- Avoid it at all cost!
  - ChatGPT etc…
- For more info: https://www.ucl.ac.uk/ioe-writing-centre/reference-effectively-avoid-plagiarism/plagiarism-guidelines

Topics

UCL

- Why Python?
- Types, variables, sequence, branching, loops
- Strings and functions. Data structures; lists, dictionaries and sets
- Object-oriented programming
- File handling and exceptions
- Data analysis using Python libraries

Learning Objectives

- Understand why Python is a useful programming language for developers
- Solve problems using a large sub-set of the language effectively
- Understand how to write simple applications
- Design object-oriented programs with Python classes
- Use programming tools such as an integrated development environment (IDE) and debugger
- Leverage the power of Python libraries in their code

Topics

UCL

- Why Python?
- Types, variables, sequence, branching, loops
- Strings and functions. Data structures; lists, dictionaries and sets
- Object-oriented programming
- File handling and exceptions
- Data analysis using Python libraries

Introduction

- *Computer scientists = mathematicians*: use formal languages to denote ideas (specifically computations)
- *Computer scientists = engineers:* design things, assembling components into systems and evaluating trade-offs among alternatives
- Computer scientists = scientists: observe the behaviour of complex systems, form hypotheses, and test predictions
- Problem solving: ability to formulate problems, think creatively about solutions, and express a solution clearly and accurately
- The process of learning to program is an excellent opportunity to practice problem-solving skills
- On one level, you will be learning to program, a useful skill by itself. On another level, you will use programming as a means to an end

Introduction to COMP0015

UCL

- What is this module about?
  - Learn how to solve problems using coding as a tool
  - We use Python as an introductory language …
  - ... but programming principles are universal
- At the end on the term, you will be able to:
  - Understand why Python is a useful programming language for problem solving
  - Be able to design and develop applications effectively
  - Be able to use appropriately the most common classes in Python Standard Library
  - Be used to tools such as an integrated development environments (IDEs)
  - Be aware of the programming and debugging processes

Programme

- What is a programming language?

  A syntax that allows you to define programs that a computer can understand

- What is a program?

  - A set of instructions telling a computer how to perform a particular task (a little like a recipe)

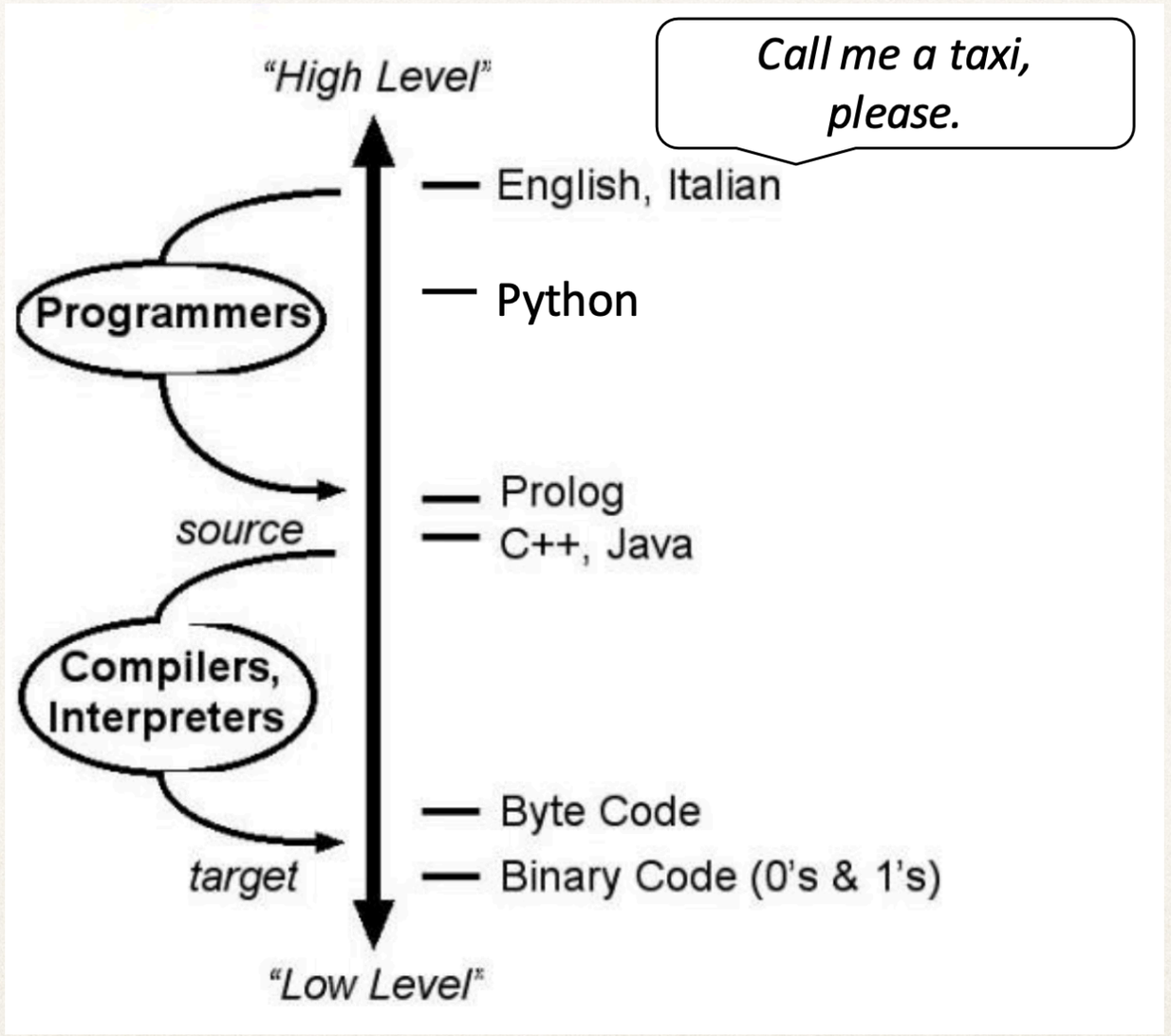
    - A computer will not do anything, unless you tell it to
    - A computer will only do whatever you tell it to
    - But it does so, very quickly and accurately

Why Python?

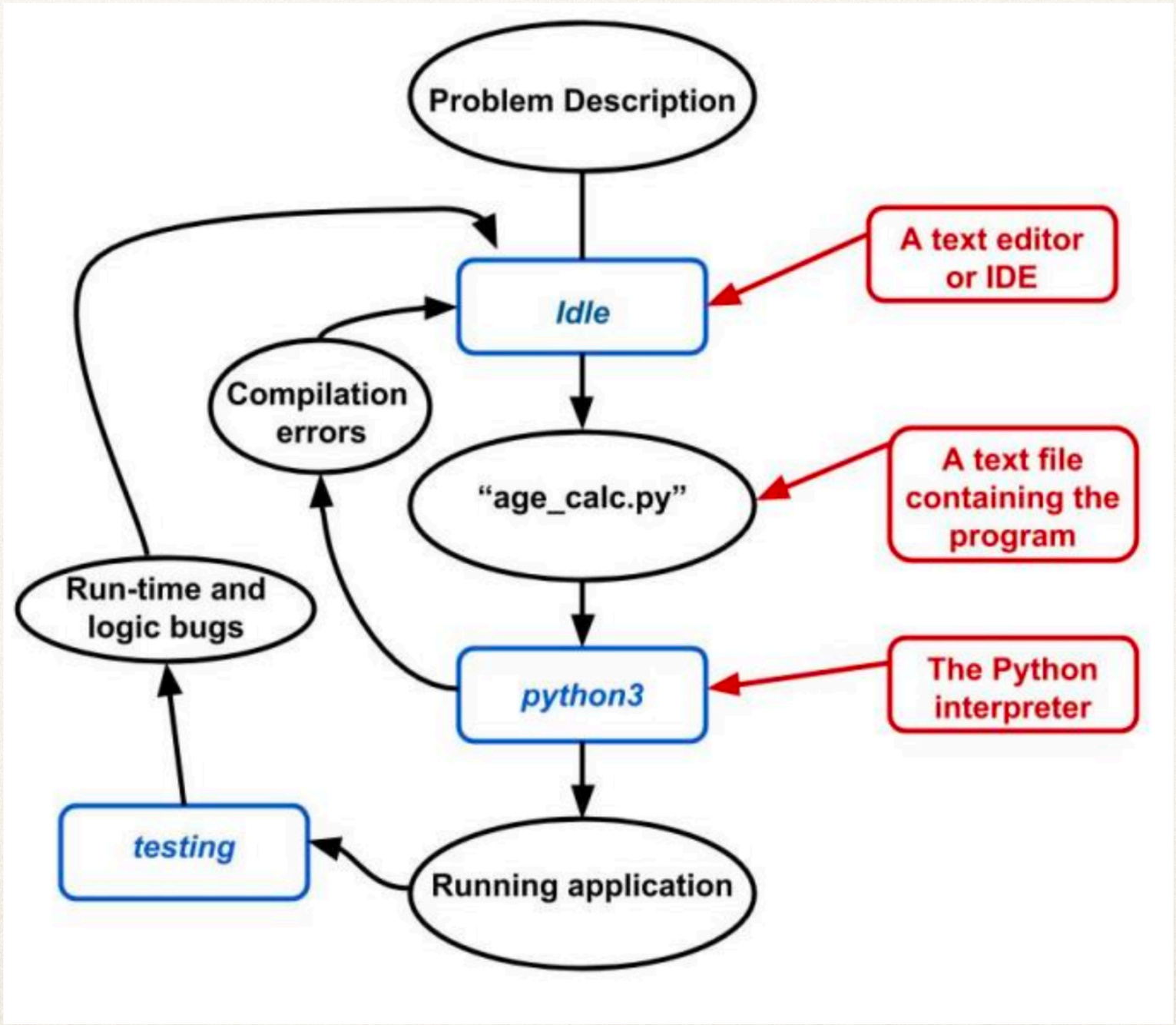
UCL

- Consistently ranked in the top ten most popular programming languages (TIOBE Programming Community Index; Language of the Year in 2007, 2010, and 2018)
- Large organisations that use Python include Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify
- Python can serve as a scripting language for web applications
- The jewel of the queen: Many libraries for machine learning —Pandas
- Libraries for other purposes:
  - Scientific Computing: *NumPy, SciPy, MatPlotLib*
  - Mathematics: *SageMath*
  - Specialised domains: *Biopython, Astropy*

One of many options



Overview


Problem Description
A text editor or IDE
Idle
Compilation errors
"age_calc.py"
A text file containing the program
Run-time and logic bugs
python3
The Python interpreter
testing
Running application

First steps

UCL

1. Set up a Python development environment (see Session materials)
2. Get writing, testing and debugging programs
3. Choose and start reading the recommended texts as soon as possible, and repeatedly re-read as many times as you find necessary for a full understanding
4. Experiment with exercises from the book, exercises from the lab sessions, and with small programs of your own design as you read
5. When facing a new problem, always think about an algorithm (sequence of steps) to solve it (before coding)
6. Repeat steps (2-5) steadily and regularly from the beginning of the course (It is impossible to learn programming in a last-minute rush! Recommended: 6 hours/week)
7. Do not be embarrassed if you get lost, or initially find programming very difficult. Ask for help!
8. Practice! Practice! Practice! (Code, Programme, Implement)

Remember

1. Check the Materials (if you did not already)
2. Get the tools. Install Python 3 and/or find out how to set up our IDE (VS CODE)
3. Go through the exercises set this week (ask questions as needed)

For next week:

Review the texts on the Reading List

Do more exercises if you need

Prepare for next sessions:

Read Materials, as they become available

Extend your knowledge, using your chosen book

Practice! Practice! Practice! (Code, Programme, Implement)

References

- Python 3 Reference Standard Library Reference, containing all the built-in facilities:
  - https://docs.python.org/3/library/index.html
- Language Reference, with all the intimate details of the language (some may be obscure until you know more about programming languages):
  - https://docs.python.org/3/reference/index.html
- The python 3 tutorial, that gives a tour of all language features:
  - https://docs.python.org/3/tutorial/index.html