Programming in Python[1]

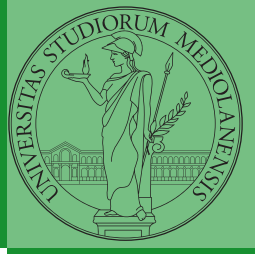
Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

Academic year 2024/25, I semester

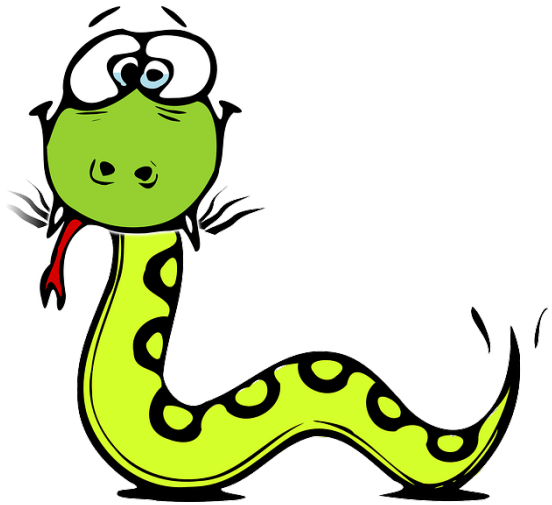
Lecture I: Programming in Python for quantitative biologists

Programming in Python (for quantitative biologists)



The course introduces imperative programming by referring to the Python language.

1. Python3 and its object-oriented features;
2. Python3 libraries that can be useful in scientific computation and data analysis, in particular `NumPy` and `pandas`.



Everything will be available on:
https://mameli.docenti.di.unimi.it/pyqb

Course schedule

- Tuesdays: 10:30 **309**, Thursday: 8:30 **B6**, Fridays: 8:30 **Delta**
- Lectures: 40h, Labs: 16h Not every week we will have three lectures, check the schedule on the web site
- Labs mainly on Friday, will start on Oct 25
- We will explore different setups: (1) a "scaffolded" one for the first steps, (2) the plain python interpreter, and finally (3) the notebooks popular in scientific practice
- Tutor: TBD (computer science master student)
- Text: every Python3 reference/book/tutorial is ok, you can access freely to the book linked on the website
- Final test: write (small) python programs without help

Attendance requirements

- For Quantitative Biology students, since 2024/25 attendance is mandatory.
- To be admitted to the exam, a QB student must have attended ≥ 10 lectures and ≥ 6 labs.
- Doing a (github) homework within a week, counts as $\frac{1}{3}$ of a lecture.

Why Python?

Programming can be approached in many "languages", the fundamental skills are general... but you cannot learn without referring to a specific language.

- A precise requirement of the teaching committee
- Very popular in the scientific landscape
- Easy to learn, many useful libraries, free software
- Alternatives: Fortran, C, Matlab, Mathematica, R, Julia, ...
- Python is slower, but it is considered easier to understand and manage

Which Python?

We will use Python3 (current version is 3.12): be careful when looking around, Python2 is still common (but deprecated) and incompatible. Python supports different "paradigms", we will focus on:

- Imperative programming: programs describe changes in *registers* and the *executing environment*;
- Object-oriented: complex (imperative) programs are organized around objects in order to hide and isolate complexity.

This is a programming course: I will try to propose example that I believe could be useful in your daily practice, but I'm not a biologist.

Programming

Programming in science can serve two (almost opposite) goals:

1. Understanding every detail of a computational process;
2. Compose computational process by assembling powerful build blocks of which you understand very little.

Most of the current popularity of programming is related to goal 2. . . with many *sorcerer's apprentices*. But this course will focus mainly on goal 1. In the last part of the course we will bend towards 2, hopefully with a solid background.

Programming can be both hard and addictive: Teach Yourself Programming in Ten Years

Fundamental concepts of Python

The programmer describes computational processes in terms of:

objects : all the entities manipulated by the program, each has an identity (can be distinguished) and a value, that is an element in a specific type (a set of values together with the operations that make sense on them)

basic types : integers (`int`), floats, strings (`str`), functions; they can be composed in more complex, user-defined, types

variables : names used to refer to objects; the same name can refer to different objects during the same process

special commands : the only way to change the execution environment (i.e., the "virtual machine" provided by the operating system) is to use system calls; syscalls change from system to system (e.g., Linux vs. Windows), but Python wraps them and they appear like the functions written by the programmers (e.g., `print`), even if they could not be programmed in Python.

Let's try!

https://python.di.unimi.it/

You can use it without any personal account, but if you want support you must create one, putting me as the "guru": `mmonga`

This platform will be used for the first lessons, since it requires no setup at all: everything happens in the browser (and the server).
(Thanks to the University of Waterloo, Canada for providing the CS Circles)