

Python Programming

Introduction

Mihai Lefter



Outline

Introduction

About Python

Running Python Code

This Course

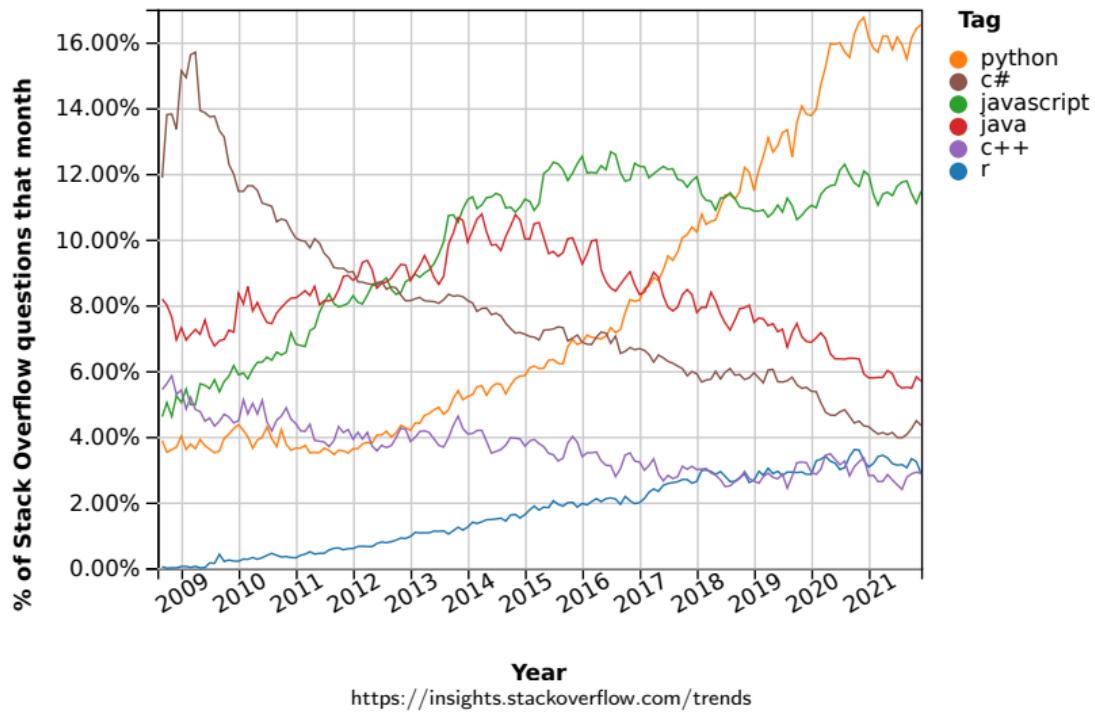
Why Python?

- Low barrier to entry.
- Widely used with a large community.
- Rich (scientific) libraries.



Introduction

Community



History

- Created early 90's by Guido van Rossem at CWI.
 - Name: Monty Python.
- Design is driven by code readability.

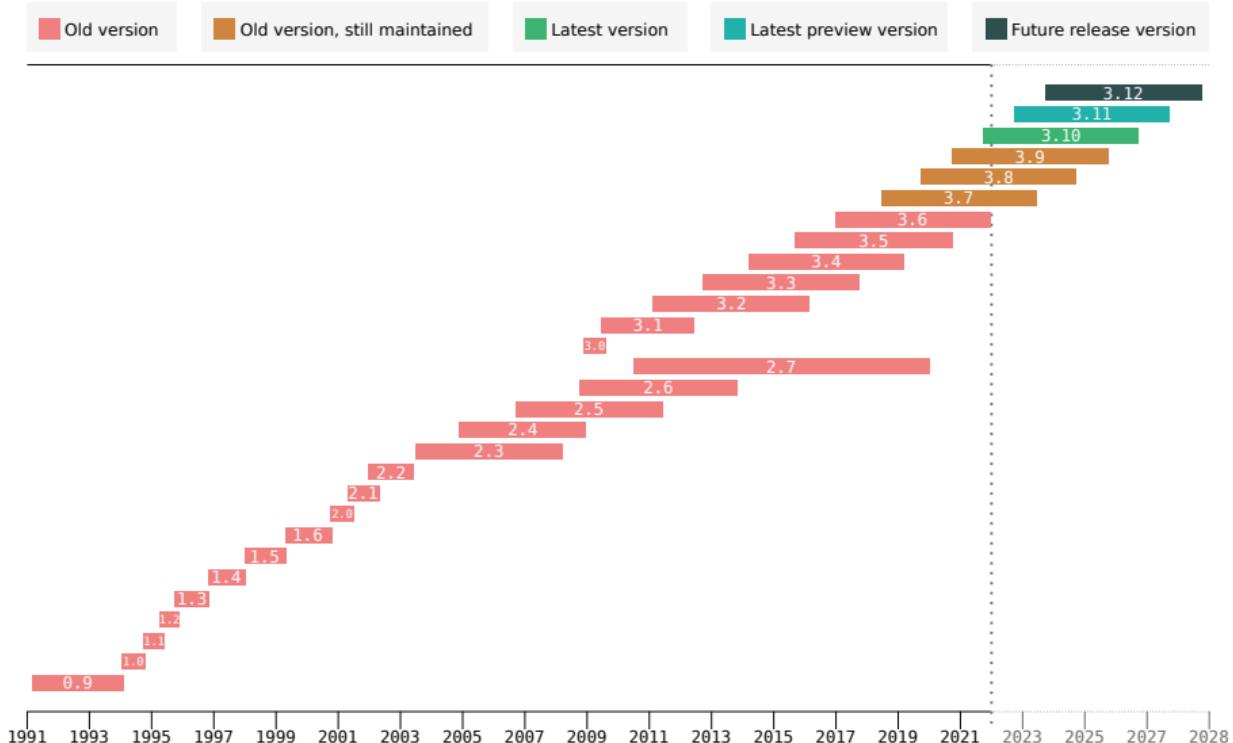


Centrum Wiskunde & Informatica



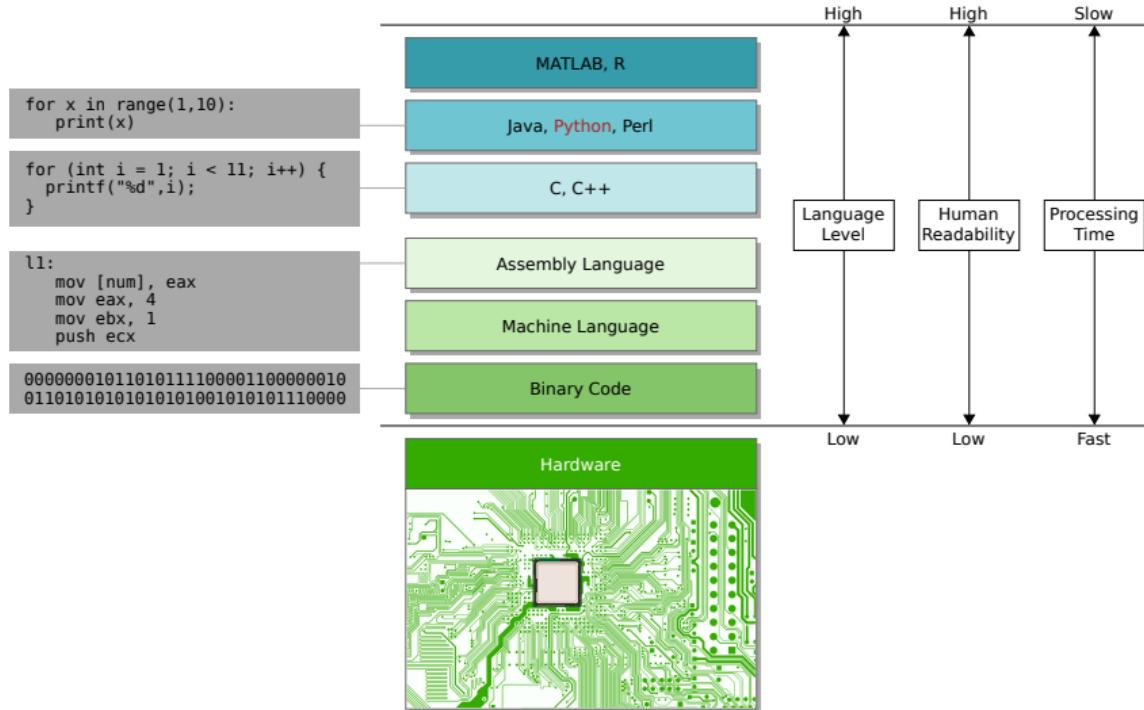
About Python

Versions



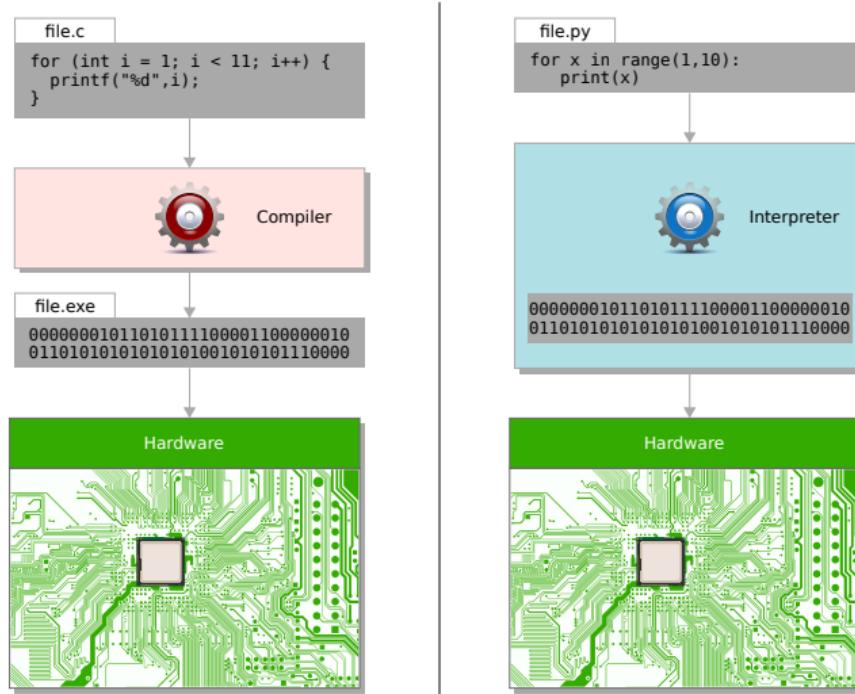
About Python

High-level programming language



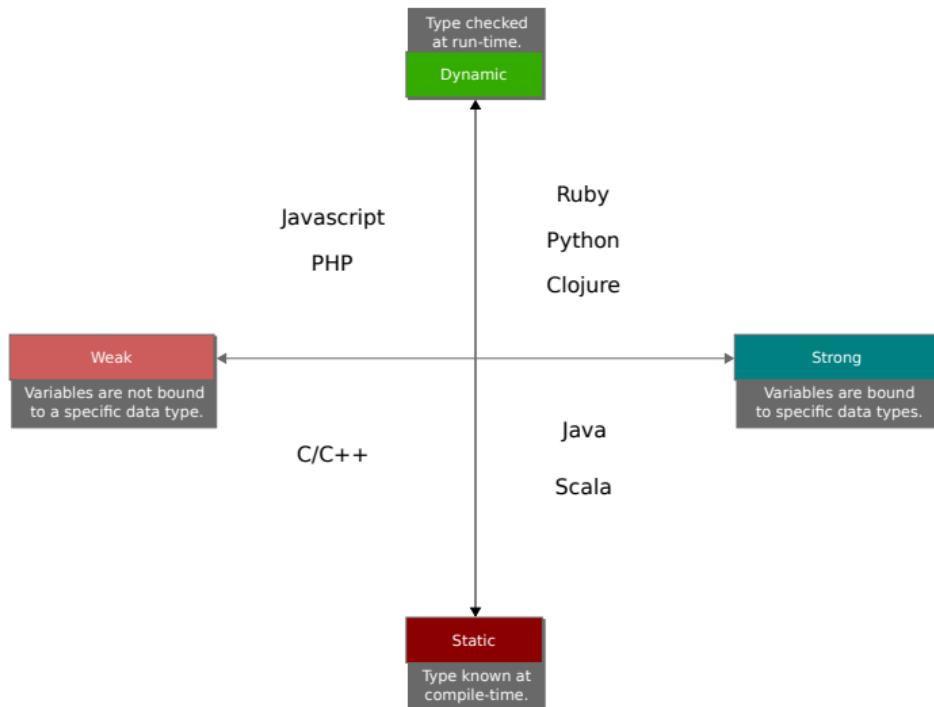
About Python

Interpreted



About Python

Dynamically and strongly typed



Other Features

- Imperative.
 - With some functional programming.
- Object-oriented.
- Automatic memory management.



Interactively:

- Statement by statement, directly in the interpreter.

Running Python Code

Interactively:

- Statement by statement, directly in the interpreter.

```
terminal
$ python
Python 3.7 (default, Nov 26 2019, 10:23:46)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or
"license" for more information.
>>> print('Hello world')
Hello world
>>>
```

Running Python Code

Interactively:

- Statement by statement, directly in the interpreter.

```
terminal
$ python
Python 3.7 (default, Nov 26 2019, 10:23:46)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or
"license" for more information.
>>> print('Hello world')
Hello world
>>>
```

- Interpreters are great for prototyping.
- But not really suitable if you want to share or release code.

Running Python Code

Interactively:

- Statement by statement, directly in the interpreter.

```
terminal
$ python
Python 3.7 (default, Nov 26 2019, 10:23:46)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or
"license" for more information.
>>> print('Hello world')
Hello world
>>>
```

Non-interactively:

- By editing a file and running the code afterwards.

- Interpreters are great for prototyping.
- But not really suitable if you want to share or release code.

Running Python Code

Interactively:

- Statement by statement, directly in the interpreter.

terminal

```
$ python
Python 3.7 (default, Nov 26 2019, 10:23:46)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or
"license" for more information.
>>> print('Hello world')
Hello world
>>>
```

Non-interactively:

- By editing a file and running the code afterwards.

first_script.py

```
1 print("Hello world!")
```

- Interpreters are great for prototyping.
- But not really suitable if you want to share or release code.

Running Python Code

Interactively:

- Statement by statement, directly in the interpreter.

terminal

```
$ python
Python 3.7 (default, Nov 26 2019, 10:23:46)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or
"license" for more information.
>>> print('Hello world')
Hello world
>>>
```

Non-interactively:

- By editing a file and running the code afterwards.

first_script.py

```
1 print("Hello world!")
```

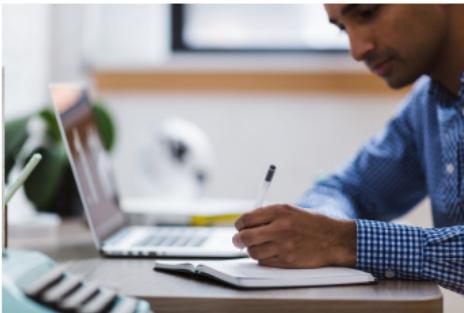
terminal

```
$ python first_script.py
Hello world
$
```

- Interpreters are great for prototyping.
- But not really suitable if you want to share or release code.

This Course

- Aimed at PhD students, Postdocs, researchers, analysts, ...
- Focus on:
 - Programming as a tool to do your research.
 - Basic understanding of Python.



Hands On!

Programming is fun!

- You only learn programming by doing it.
- Lecture format:
 - Blended teaching + exercising.
- Repeat the code from the slides/lectures and play around with it.
- Do the session exercises.
- Practical sessions.



This Course

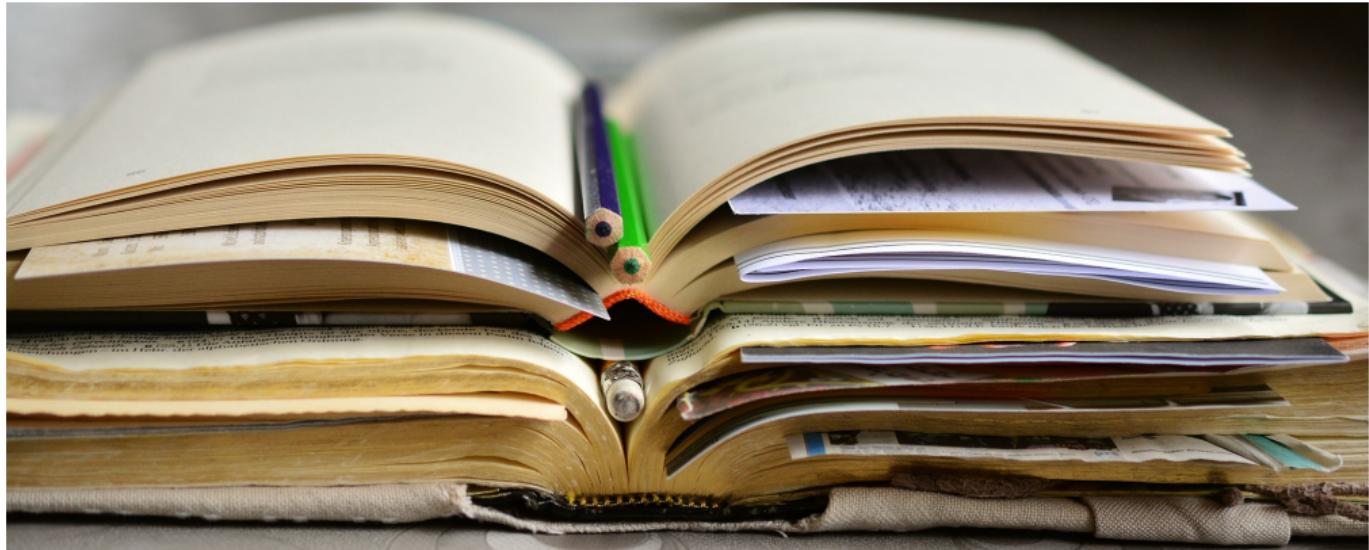
Program

Day 1		Day 2		Day 3		Day 4	
9:30	30 min	Introduction	30 min	Assignments review	30 min	Assignments review	30 min
10:00	50 min	Data Types	50 min	Functions	50 min	String methods, errors and exceptions	50 min
10:50	10 min	Break	10 min	Break	10 min	Break	10 min
11:00	1 h	Flow Control	1 h	Object-oriented programming	1 h	Standard library, reading and writing files	1 h
12:00	30 min	Lunch break	30 min	Lunch break	30 min	Lunch break	30 min
12:30	2.5 h	Practice	2.5 h	Practice	2.5 h	Practice	2.5 h
15:00							
15:30							

This Course

Material

<https://git.lumc.nl/courses/programming-course/-/blob/master/README.md>



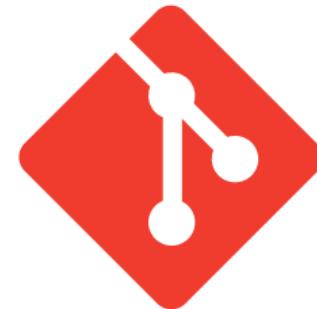
Practical Sessions

- We make use of GitHub Classroom.
 - GitHub account required.
 - Receive link with assignment repository.
- Own forked repository to work on:
 - Clone it.
 - Code it.
 - Push it.
- Direct file upload to repository is also possible.



Software requirements

- Anaconda:
 - Python 3.x.
 - Comes with all that's required:
 - Python interpreter.
 - Jupyter Notebook.
 - Libraries: NumPy, Panda, matplotlib, Bokeh, Biopython, ...
 - [Installation instructions.](#)
- Git:
 - [Installation instructions.](#)



Getting help

- Ask a question.



Acknowledgements

Martijn Vermaat

Jeroen Laros

Jonathan Vis

