Introduction to Programming with Python

Dr. Anatol Wegner

Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilians-Universität Würzburg
Würzburg, Germany

anatol.wegner@uni-wuerzburg.de

Lecture 01
Introduction to Python

October 25, 2024

What is programming?

1. the process creating an instructional program for a device to make it perform a certain task.
2. attempting to get a computer to complete a specific task without making mistakes.
3. the process of creating precise set instructions for a computer to perform a well defined task without mistakes.

- modern computers work with (binary) machine-level code to encode data and instructions
- higher level programing languages allow us to writte programs that are closer to human language (usually English)

Macbook computer

image credit: Wikipedia, VladIsLoveW, CC-BY-SA

Programming languages

- just as natural language, programing languages have rules and conventions, synthax and semantics
  - a set of **precise/formal** rules (grammar) that define what is valid code and/not
- compilers and interpreters are programs that translate higher level programming languages to machine-code

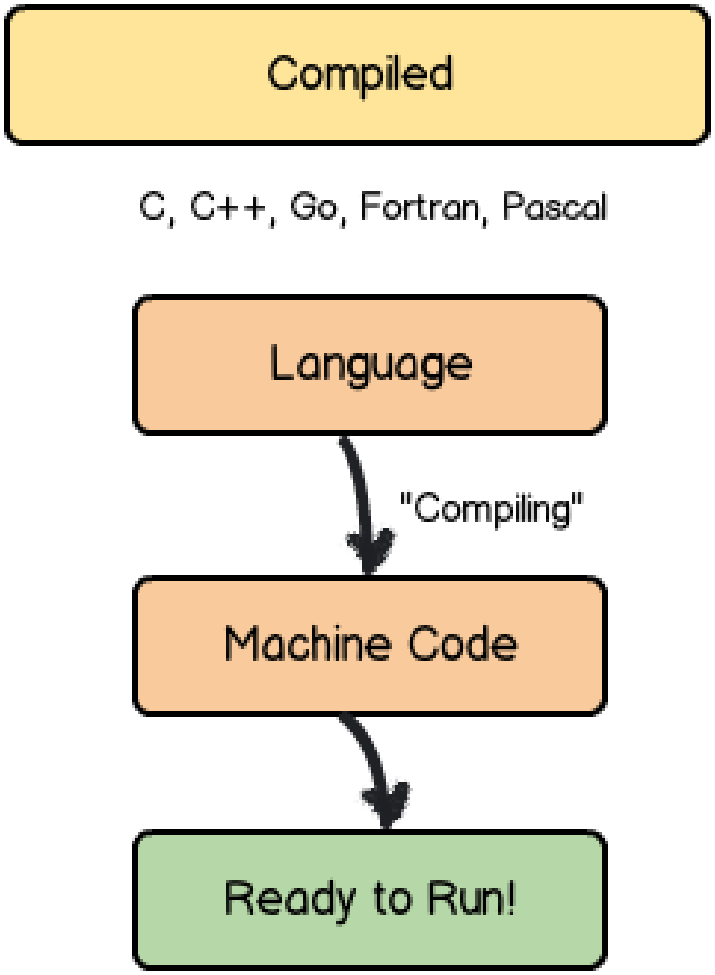
Some popular computer languages

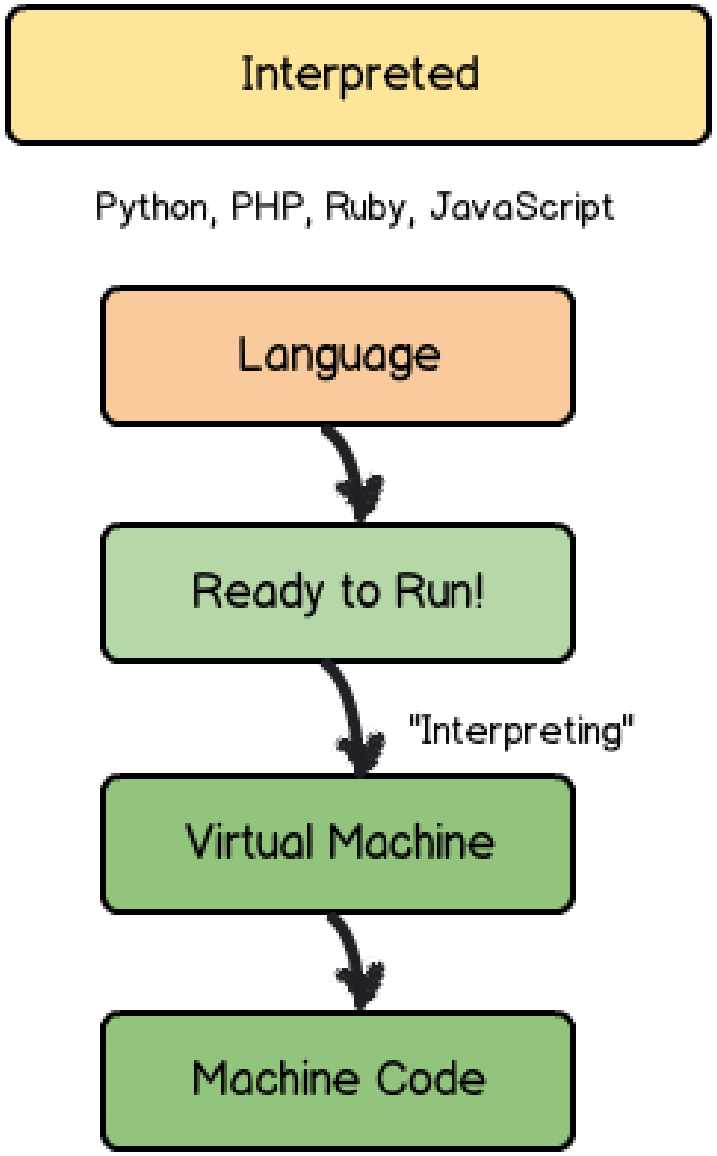
Programming languages

- just as natural language, programing languages have rules and conventions, synthax and semantics
  - a set of **precise/formal** rules (grammar) that define what is valid code and/not
- compilers and interpreters are programs that translate higher level programming languages to machine-code
- there are many programming languages: choices of programming language depends on personal preference and application.

Some popular computer languages

Compiled vs Interpreted languages





Compiled vs. Interpreted Languages

- **compiler translates program** in high-level language to machine code **before** it can be executed
  - compiled binaries are not portable
  - users may need to compile source code
  - each change requires recompilation

Compiled vs. Interpreted Languages

- **compiler translates program** in high-level language to machine code **before** it can be executed
  - compiled binaries are not portable
  - users may need to compile source code
  - each change requires recompilation
- **interpreter directly executes instructions** in high-level programming language

Definition

An **interpreter** is a program that directly executes instructions written in a programming language, without requiring its prior compilation to machine code.

Compiled vs. Interpreted Languages

- **compiler translates program** in high-level language to machine code **before** it can be executed
  - compiled binaries are not portable
  - users may need to compile source code
  - each change requires recompilation
- **interpreter directly executes instructions** in high-level programming language
- no need for (re)compilation, no non-portable binaries

Definition

An **interpreter** is a program that directly executes instructions written in a programming language, without requiring its prior compilation to machine code.

Compiled vs. Interpreted Languages

- **compiler translates program** in high-level language to machine code **before** it can be executed
  - compiled binaries are not portable
  - users may need to compile source code
  - each change requires recompilation
- **interpreter directly executes instructions** in high-level programming language
- no need for (re)compilation, no non-portable binaries
- interpreted languages are **typically slower than compiled languages** (but not necessarily)

Definition

An **interpreter** is a program that directly executes instructions written in a programming language, without requiring its prior compilation to machine code.

Python

- `python` is the most **popular interpreted programming language**
- widely-used for **data processing, analytics, and machine learning**
- object-oriented, dynamically typed, automatic memory management

Python

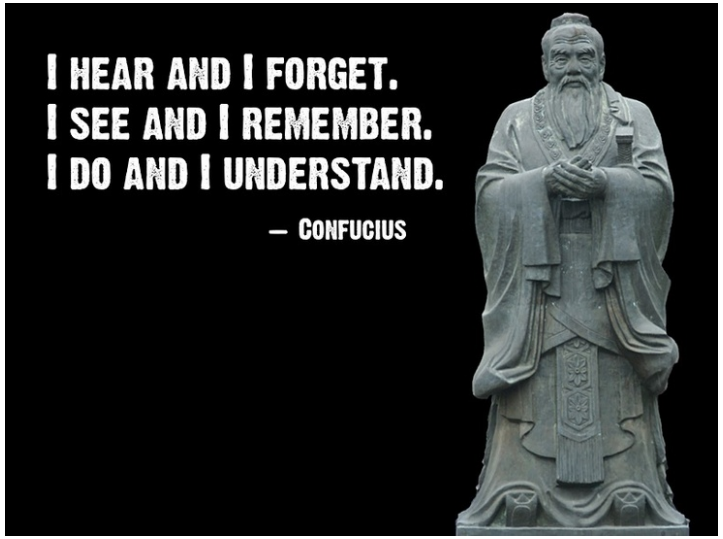
- `python` is the most **popular interpreted programming language**
- widely-used for **data processing, analytics, and machine learning**
- object-oriented, dynamically typed, automatic memory management
- user-friendly, great for **beginners in programming**
- **rich ecosystem of libraries** (modules) that implement almost any imaginable functionality

Guido van Rossum, developer of `python`

image credit: Wikpedia, Doc Searls, CC BY-SA 2.0

Course structure

- Programming languages are best learned trough practice.
- Lectures:
  - Introduce new concepts,
  - Practical examples,
  - Practice session.

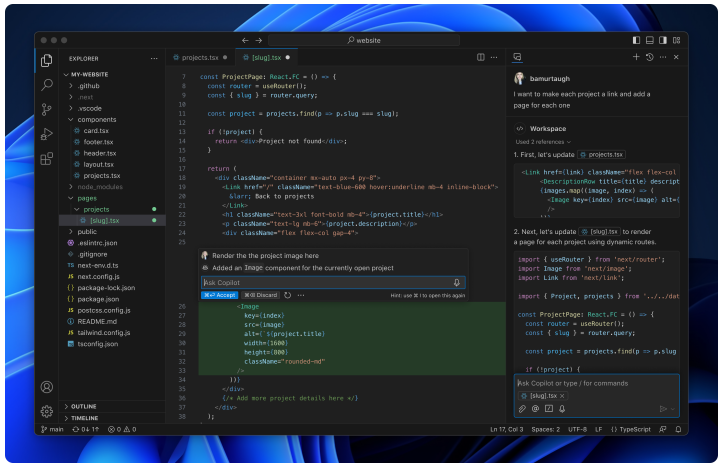


Getting Python

- Python is already part of MacOS and Linux
- For Windows:
  - recommended: anaconda.com/download
  - alternatively: python.org/downloads/windows/

Integrated Development Environment (IDE)

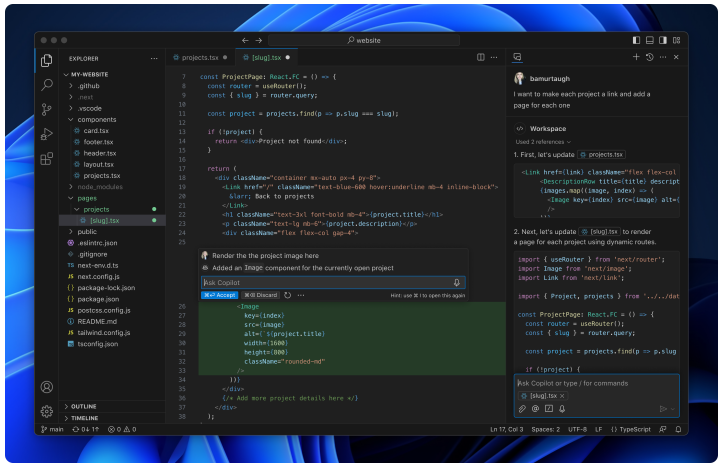
- Python code can be written with any text editor.
- IDEs offer a convenient way of writing code:
  - automatic annotation,
  - debugging,
  - code execution,
  - + tools and extensions...
- Visual Studio Code
  - app store or https://code.visualstudio.com/
  - once installed install Python extension by Microsoft



The VS code IDE

Integrated Development Environment (IDE)

- Python code can be written with any text editor.
- IDEs offer a convenient way of writing code:
  - automatic annotation,
  - debugging,
  - code execution,
  - + tools and extensions...
- Visual Studio Code
  - app store or https://code.visualstudio.com/
  - once installed install Python extension by Microsoft



The VS code IDE

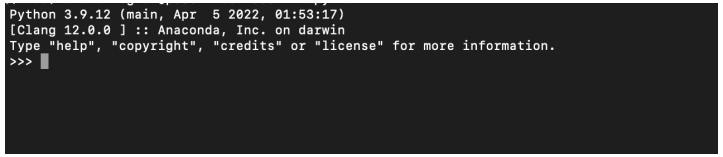
Practice Session 1

- Setting up Python
- Setting up VS code

Interacting with Python

Interactive Python shell:

- open terminal and type ’Python’
- can execute Python commands
- code is executed *sequentially*

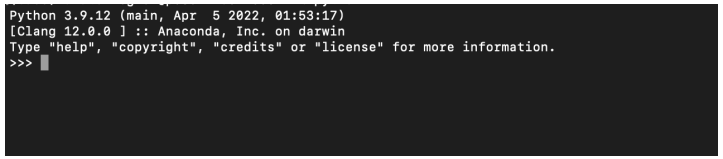


The Python shell

Interacting with Python

Execute a Python (.py) file:

1. write code using any text editor,
2. save with extension .py (e.g Hello.py)
3. run using »Python Hello.py
4. will run the code line by line.

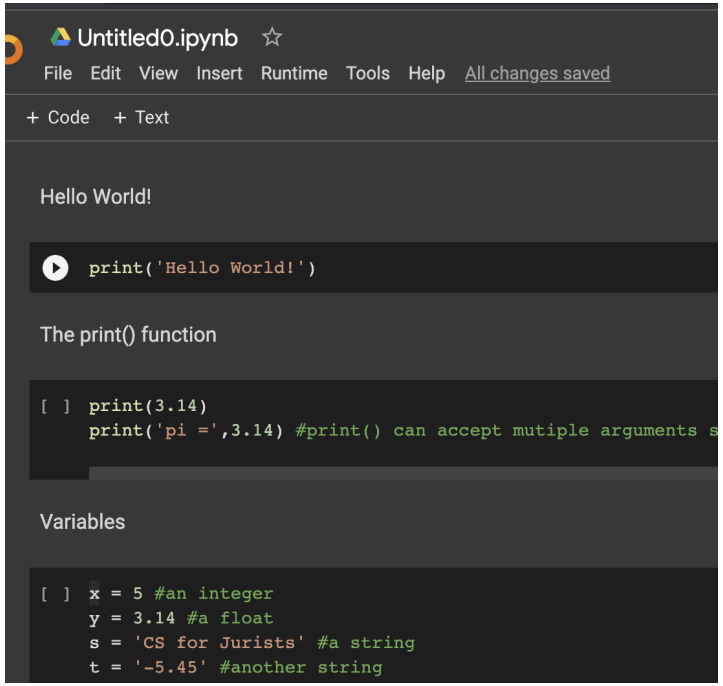


The Python shell

Interacting with Python

Jupyter notebooks:

- interactively edit & execute code
- code is executed in blocks
- display visualizations (e.g graphs, images...)
- text blocks in MarkDown.
- we will mostly use Jupyter notebooks...
- see https://colab.google/ for a free online Jupyter server (requires google account)



A jupyter notebook

Practice Session 2

- Running our first Python program
- Basics of Jupyter notebooks

Strings

- In Python text values are called strings
  - strings are just lists of characters
  - strings are defined using quotation marks (single or double)
- Examples:
  - a='Hello world' (a string)
  - b='3.14' (another string)
  - c=3.14 (not a string)

Basic operations

- Arithmetic operations (+, -, *, /, **, %):
  - x**y (x^y): 3**2 = 9
  - % (mod) : 3 % 2 = 1
- Comparisons :
  - 3==4 (False)
  - 3!=4 (True)
  - 3<4 (True)
  - 3>4 (False)
  - 3<=4 (True)
- Logical operations (and, or, not):
  - False and True (False)
  - False or True (True)
  - False or not True (False)

Warning 1

Not all operations work with all types

Warning 2

Output of operations depend on types

- 'Alan'+'Turing'='AlanTuring'
- 3 + True = 4
- True + True = 2
- 'Alan'+3=? (error)

Basic operations

- Arithmetic operations (+, -, *, /, **, %):
  - x**y (x^y): 3**2 = 9
  - % (mod) : 3 % 2 = 1
- Comparisons :
  - 3==4 (False)
  - 3!=4 (True)
  - 3<4 (True)
  - 3>4 (False)
  - 3<=4 (True)
- Logical operations (and, or, not):
  - False and True (False)
  - False or True (True)
  - False or not True (False)

Warning 1

Not all operations work with all types

Warning 2

Output of operations depend on types

- 'Alan'+'Turing'='AlanTuring'
- 3 + True = 4
- True + True = 2
- 'Alan'+3=? (error)

Practice Session 3

- arithmetic operations
- logical operations

You can find the notebooks for the lectures at `https://gitlab2.informatik.uni-wuerzburg.de/ml4nets_notebooks/2024_wise_infhaf_notebooks`

In summary

- Basics of programming
- Compiled vs interpreted programming languages
- Overview of the Python programming language
- Setting up Python and IDE
- We wrote our first program in Python
- We learned about **basic variable types and operations**

Self-study questions

1. What is the difference between a compiled and an interpreted programming language?
2. What is an integrated development environment and what are its advantages?
3. What is synthax and why is it important?
4. What is a string?
5. List some of the mathematical operations in Python.
6. can you write down the truth table of ’or’ and ’and’?

Literature

reading list

- F Kaefer, P Kaefer: **Introduction to Python Programming for Business and Social Science Applications**, SAGE Publications, 2020
- **Official Python documentation** https://docs.python.org/
- **Python tutorial**: https://docs.python.org/3/tutorial/