Technische Universität Braunschweig

Institute for Computational Modeling in Civil Engineering

Algorithms & Programming

Lecture 1

Prof. Dr.-Ing. Henning Wessels

Are you ready?

A. Yes, let‘s go!
B. Wait a second…
C. I need help!
D. Bye bye – I‘ll watch the recording later

slido.com with **#076211**

What is your field of study?

A. Civil Engineering
B. Environmental Engineering
C. CSE
D. Other

How much experience do you have with programming?

A. No experience at all
B. Experience from school
C. Experience from university (e.g. „Einführung in die Programmierung“)
D. I am a passionate hobby programmer

What do you expect to learn?

Programming Practice

Your boss wants you to write a program, that ….

Eventually, your program will be a subprogram of another subprogram of …

Somebody else did sth. similar before and open-sourced a solution ….

Research Software Engineering

- Coding infrastructure: https://irmb.gitlab-pages.rz.tu-bs.de/knowledge-base/content/intro.html
- The Suresoft project: https://www.tu-braunschweig.de/suresoft

What are you going to learn

- Basics of programming in Python
- Basics of research software engineering
- Unified Modeling Language (UML) and Object Oriented Programming (OOP)
  - Application to Finite Element Method
  - Application to Machine Learning
- Data structures
- Complexity of Algorithms
- Recursion

Outline

- **Organizational Matters**
- Basics of Programming (in Python)

Examination

- Written examination (60min)
- Workload
  - Civil Engineering: 3 ECTS
  - CSE: 8 ECTS
    - 2 (graded) assignments

CIP Pool

- Access during lectures and to study on your own during free time slots
- Registration and schedules:

  https://www.tu-braunschweig.de/irmb/lehre/computerpools/belegungsplaene

Literature and Secondary Information

Where to find help:

- studIP: slides, exercises
- https://www.learnpython.org

Getting Started – Install Python

- Download the latest Python version:
  - https://www.python.org/downloads/

- Download Miniconda
  - Helps you to organize different python versions required for your programs
  - https://docs.conda.io/en/latest/miniconda.html

Getting Started – Install the Editor

Visual Studio Code

VS Code is an integrated development environment (IDE) for developing in Python and other programming languages

- Free download at: https://code.visualstudio.com/
- Python Extension: https://marketplace.visualstudio.com/items?itemName=ms-python.python

Outline

- Organizational Matters
- **Basics of Programming (in Python)**

Basics of programming

Computer programming

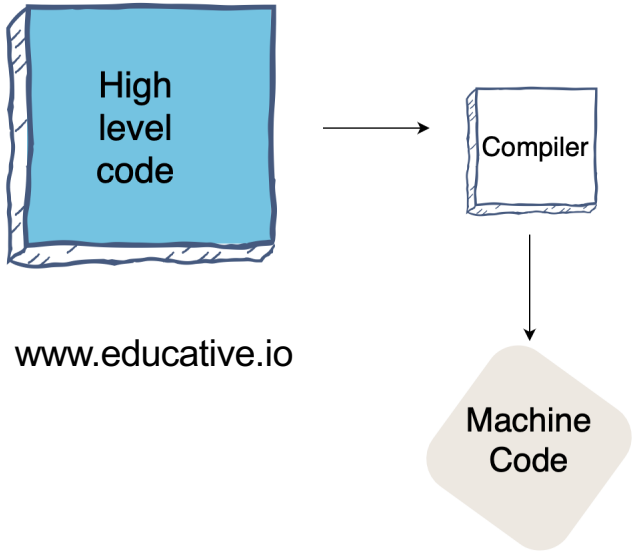
- is the process of designing, writing, testing, debugging, and maintaining the source code of computer programs.

- A program is sequence of instructions for solving a (formally well-defined) problem.

Basics of programming

Program execution for compiled languages

- Central Processing Unit(s) (CPU) execute(s) digital machine code (0/1-combinations)
- Human readable programs (source code) must be translated (compiled) into machine code before execution
- Compiled languages are C, C++, FORTRAN, …



www.educative.io

Basics of programming

Interpreted languages

- Are not translated to machine code
- The source code is *read (interpreted)* and *executed* by another program called the *Interpreter*
- Python is an interpreted language

Compiled vs. interpreted languages

Advantages of compiled languages

- All compiled programs are faster as compared to any interpreted code. This is because the code does not need to be compiled while the program is running.
- A compiler gives a list of all the compilation errors during compilation. A programmer can fix the errors and execute the code again.

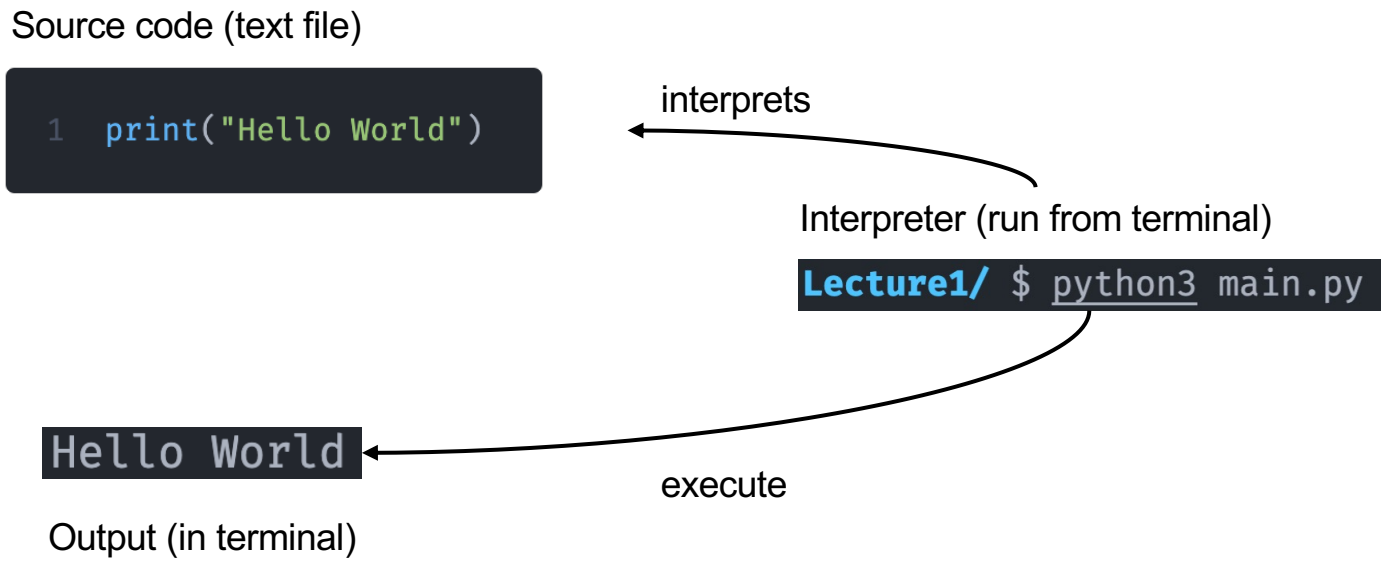
Disadvantages of compiled languages

- The entire code needs to be compiled before testing. This increases the overall run time of execution.
- The machine code depends on the platform it is running.

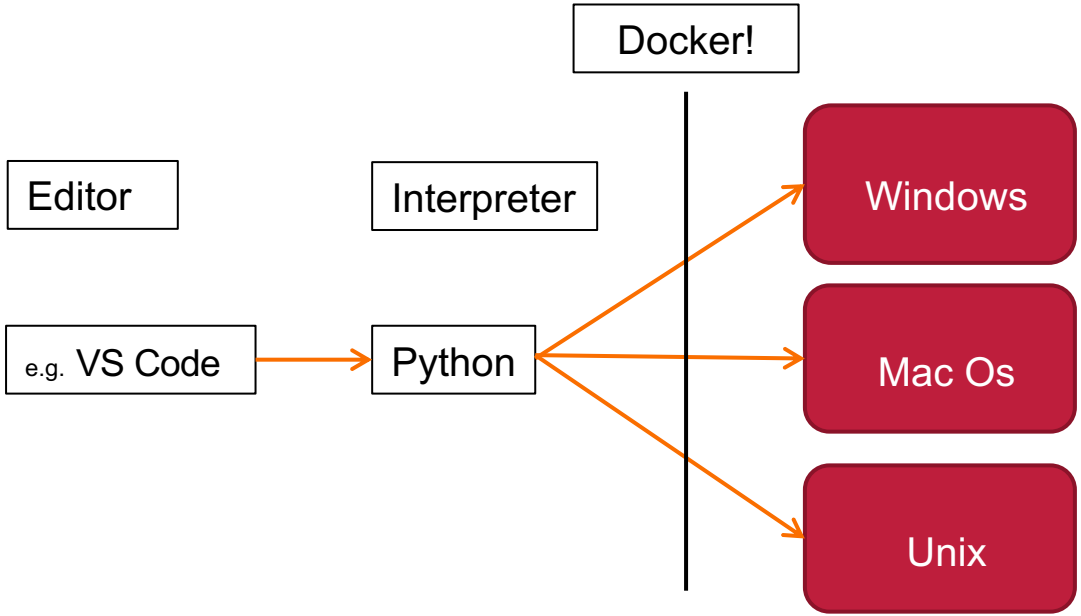
General steps of programming

- Create a source code of your program (**implementation**)
  Tool: Editor/IDE
- For compiled languages: Compile source code to generate machine code (**compilation**)
  Tool: Compiler
- Run your program (**execution**)
  Tool: Operating System (OS) / Run-Time System

Programming: interpretation & execution



Basics of programming

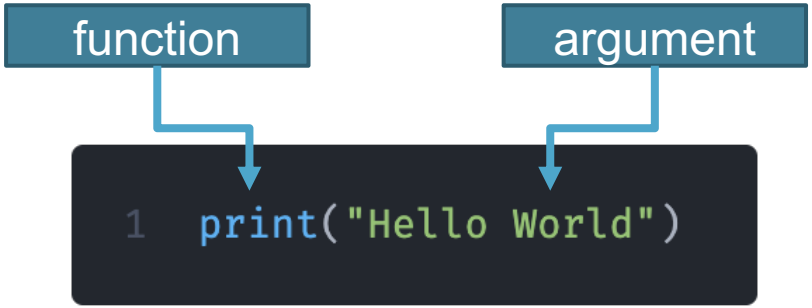


Python: your first program

- Problem: print a message “Hello World!” in terminal window
- Solution in Python syntax:

```
1  print("Hello World")
```

Python Programming



Errors, also known as “Bugs”

We distinguish between the following **types of errors:**

- **Syntax errors**: errors due to the fact that the syntax of the language is not respected.
- **Semantic errors:** errors due to an improper use of program statements.
- **Logical errors:** errors due to the fact that the specification is not respected.

Syntax errors

Syntax errors are due to the fact that the syntax of the Python language is not respected.

- Example 1: Errors in expressions:

  x = (3 + 5 *// missing closing parenthesis ')'*

  y = 3 + * 5 *// missing argument between '+' and '*'*

- Example 2: Wrong indentation

  if a > b:
  print("a is greater")

Semantic errors

Semantic errors indicate an improper use of Python statements.

- Example 1: Use of a non-initialized variable:
  i + 1 *# name ‘i' is not defined*

- Example 2: Errors in expressions:
  s = “test”
  a = 5 - s *# unsupported operand type(s) for -: 'int' and 'str'*

Logical errors

Logical errors are caused by the fact that the software specification is not respected. The program is compiled and executed without errors, but does not generate the requested result.

- Example 1: Errors in the performed computation:

```
def sum(a: int, b: int) -> int:
    return a - b ;

# this method returns the wrong value
# the specification that requires to sum two integers
```

Which type of error was made?

```
1   s = "This is a string
```

Interpreter message:

EOL while scanning string literal

A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

Which type of error was made?

```
1   s = "This is a string
```

Interpreter message:

EOL while scanning string literal

A. Syntax Error

B. Semantic Error

C. Logical Error

D. No Idea

Which type of error was made?

```
1   i = int("string")
```

Interpreter message:
invalid literal for int()

A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

Which type of error was made?

```
1   i = int("string")
```

Interpreter message:
invalid literal for int()

A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

Which type of error was made?

```
1   print "Hello World"
```

Interpreter message:

Missing parentheses in call to 'print'

A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

Which type of error was made?

```
1   print "Hello World"
```

Interpreter message:

Missing parentheses in call to 'print'

A. Syntax Error

B. Semantic Error

C. Logical Error

D. No Idea

Which type of error was made?

The following code compiles without errors and is supposed to calculate the surface area of a circle:

```
1  def area_of_circle(radius: float) -> float:
2          return 2.0 * math.PI * radius
```

A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

Which type of error was made?

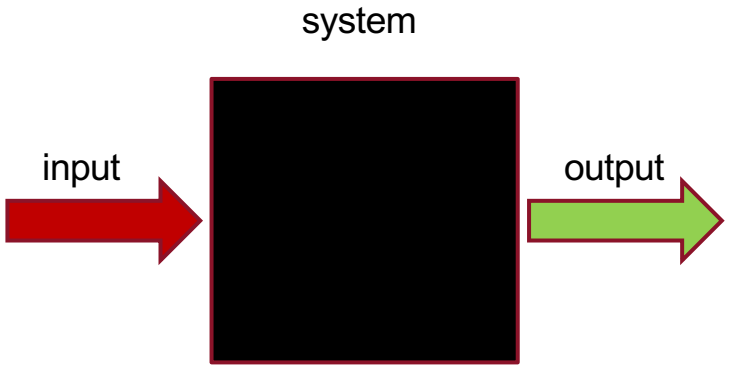
The following code compiles without errors and is supposed to calculate the surface area of a circle:

```
1  def area_of_circle(radius: float) -> float:
2          return 2.0 * math.PI * radius
```

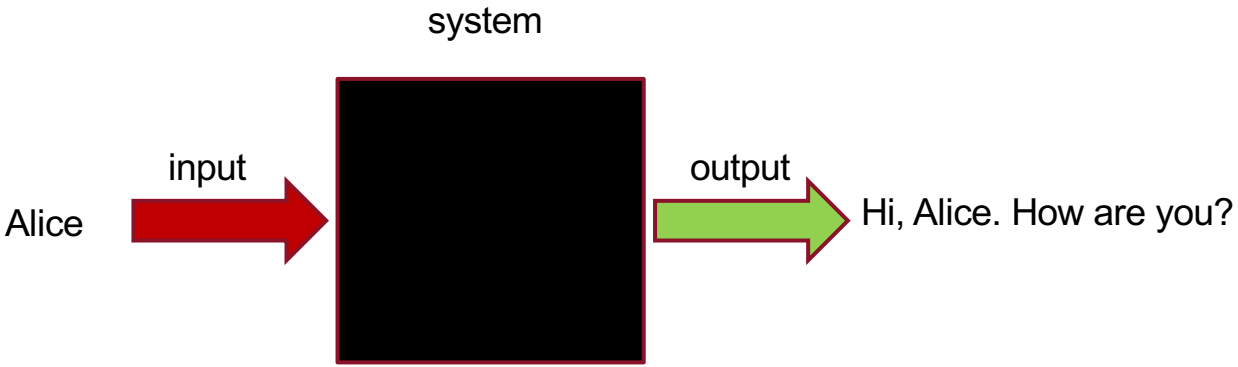
A. Syntax Error
B. Semantic Error
C. Logical Error
D. No Idea

```
1  def area_of_circle(radius: float) -> float:
2              return math.PI * radius ** 2
```

Command Line Arguments



Command Line Arguments



Command Line Arguments

Using a command-line argument

```python
import sys

args = sys.argv
name = args[1]
print(f"Hello {name}. How are you?")
```

```
Exercise/ $ python3 solutions/exercise01_task_3.py Alice
 Hello Alice. How are you?
```

Questions?

A. Yes
B. No
C. Of course!
D. I did not understand the question.