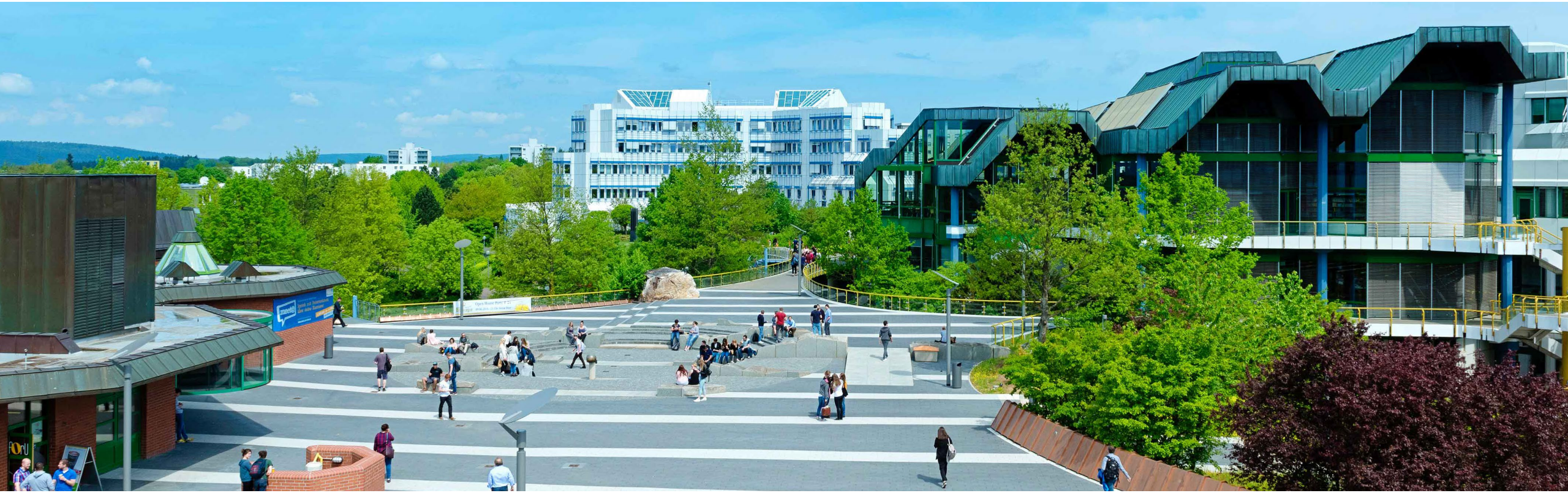
UNIVERSITÄT
TRIER



Algorithmische Methoden

Dr. Simon Werner

UNIVERSITÄT TRIER

1. Who am I?

- Dr. Simon Werner
- Researcher, lecturer, study advisor (NLP and STeM)
- Contact: werners@uni-trier.de
- Office: B319 (no appointment – no talk)
- Research interests: AI Alignment, Representation Learning, Knowledge Graph Embeddings, Large Language Models

UNIVERSITÄT TRIER

2. Who are you?

- Students of Media Science, NLP, STeM?
- First and Third semester Bachelor?
- First and Second semester Master?
- Technical background?
- Something else?

3. Will I fail/pass the exam?

- Last year:
  15 out of 86 failed on their first attempt
  19 out of 19 passed on their second attempt

  ⇒ Everyone passed eventually

  No significant difference between Subjects:
  ~ 10 % of STEM Students failed on their first attempt

  ~ 16 % of MEWI Students failed on their first attempt

  ~ 13 % of NLP Students failed on their first attempt

  ⇒ Attend the classes, listen, make notes, ask questions and prepare for the exam!

4. Schedule

2025 – Programming fundamentals	
15.10.	Introduction
22.10.	Variables and Data Types I
29.10.	Data Types II
05.11.	Loops and Conditions
12.11.	Functions
19.11.	Recursion
26.12.	Classes
03.12.	Utility Constructs
10.12.	Modules
17.12.	Exceptions

UNIVERSITÄT TRIER

4. Schedule

2026 – Applications for NLP and Media Science	
14.01.	Web Scraping
21.01.	File Operations
28.01.	Natural Language Processing
04.02.	Data Visualization
11.02.	Recap
17.02. (Tuesday!!!)	Exam

UNIVERSITÄT TRIER

5. Exercise

- 2 slots available
  - Wednesday 12:00 (Lecturer: Simon Werner)
  - Friday 10:00 (Lecturer: Christoph Hau)

- Wednesday will be taught in ENGLISH
- Friday will be taught in GERMAN

- Will start next week!

- All course material (lecture and exercise) will be uploaded to Studip

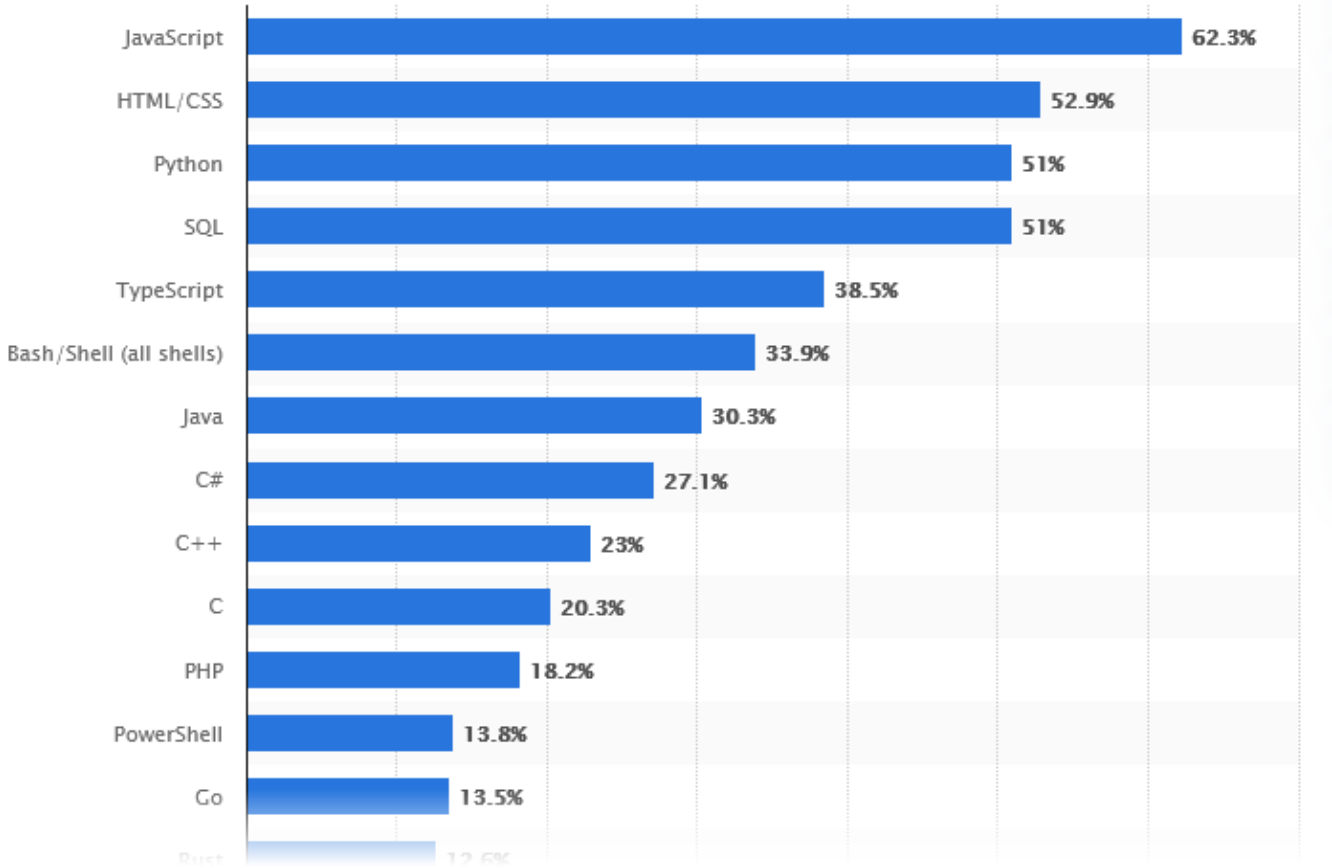
- There will be weekly homework (mostly programming exercises)

6. What is Python?

- „Python is an **interpreted**, **object-oriented**, high-level **programming language** with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, **easy to learn syntax** emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed[…]“ (https://www.python.org/doc/essays/blurb/)

UNIVERSITÄT TRIER

7. Why Python?



https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/

UNIVERSITÄT TRIER

7. Why Python?

- Web and Software Development
- **Task Automation**
- **Machine Learning and Data Science**
- Financial Analysis
- **Artifical Intelligence**

(according to: https://github.blog/developer-skills/programming-languages-and-frameworks/why-python-keeps-growing-explained/)

8. Getting Started

1. Option: Local installation
   - Download the latest Python Interpreter (https://www.python.org/downloads/)
   - Install
   - Additional installation of a suitable editor or IDE is recommended
   - Recommendation: VSCode or PyCharm (Many alternatives are available, depending on your personal preferences)

2. Option: Online tools
   - https://colab.research.google.com/
   - No local installation necessary!

9. Next steps?

- Set up your preferred Python environment and run your first own program
- Next week's exercise sessions are intended to provide help and guidance for this process
- Regular exercises from next week onwards

10. What about AI?

Consider a list x and fill it with values from 1 - 10.

Now create a second list that adds the number 5 to each value from list x.

Gemini

```
−
+# Create the first list with values from 1 to 10
+x = list(range(1, 11))
+print("List x:", x)
+
+# Create the second list by adding 5 to each value from list x
+y = [val + 5 for val in x]
+print("List y:", y)
```