# Introduction - Programming

## Programming: Everyday Decision-Making Algorithms

Dr. Tobias Vlćek

Kühne Logistics University Hamburg - Winter 2025

# About this Course

## Teaching Team



Figure 1: Dr. Tobias Vlcek



Figure 2: Dr. Nils Roemer

## About me

- Field: Optimizing and simulating complex systems
- Languages: of choice: Julia, Python and Rust
- Interest: Modelling, Simulations, Machine Learning
- Teaching: OR, Algorithms, and Programming
- Contact: [vlcek@beyondsimulations.com](mailto:vlcek@beyondsimulations.com)

. . .

> 💡 Tip
>
> We really appreciate active participation and interaction!

## Agenda

- About the course, team, and expectations
- How to learn programming (mindset and resources)
- Setting up Python with uv and VS Code

- Using notebooks with uv
- Q&A and next steps

## Course Outline
- I: Optimal Stopping
- II: Explore & Exploit
- III: Caching
- IV: Scheduling
- V: Randomness
- VI: Computational Kindness

## Participation
- Try actively participating in this course
- You will find it much (!) easier and more fun
- Lecture based on the book Algorithms to live by[1]
- Material and slides are hosted online: beyondsimulations.github.io/Programming-Everyday-Decisions

## Teaching
- Lecture: Presentation and discussion of algorithms related to everyday decision-making
- Tutorial: Step-by-step assignments to be solved and discussed together in groups
- Difficulty: Strongly depends on your background and programming experience

. . .

> 💡 Tip
>
> No worries, we will help you out if you have any questions!

## Passing the Course
- Pass/fail course without exams
- 75% attendance required for passing the course
- Hand in the assignments of at least two lectures
- Short presentation and discussion at the end
- You work together in groups of three students

## Handing in Assignments
- Each student group submits one solution
- Provide us all working notebooks of the lecture
- Hand in is due at the beginning of the next lecture
- At least 50 % have to be correct to pass
- You have to pass at least twice

---

[1]Christian, B., & Griffiths, T. (2016). Algorithms to live by: the computer science of human decisions. First international edition. New York, Henry Holt and Company.

. . .

> 💡 Tip
>
> This is just in order to provide you with working solutions after each deadline.

## Learning Python

We will mostly not cover Python during the lectures!

. . .

Question: Anybody know why?

. . .

- In our experience, the best way to learn is by doing!
- Here, we will focus on decision-making algorithms
- You will learn Python by doing the tutorials

. . .

> 💡 Tip
>
> Don't worry, we will help you out if you have any questions!

## Difficulty of the Course

- At first it might be a little bit overwhelming
- Programming is similar to learning a new language
- First, you have to get used to it and learn words
- Later, you'll be able to apply it and see results
- Important: Practice, practice, practice!

## Goals of the Course

- Learn the basics of programming
- Learn about algorithmic thinking
- Be able to apply methods and concepts
- Solve practical problems with algorithms

> 💡 Tip
>
> We are convinced that this course will be quite interesting and teach you more for your daily life than most other courses!

## Why Python?

- Origins: Conceived in late 1980s as a teaching and scripting language
- Simple Syntax: Python's syntax is mostly straightforward and very easy to learn
- Versatility: Used in web development, data analysis, artificial intelligence, and more

- Community Support: A large community of users worldwide and extensive documentation

### Help from AI

- You are allowed to use AI in the course, we use it as well (e.g., Claude, ChatGPT, LLama3 …)
- These tools are great for learning Python!
- Can help you a lot to get started with programming

. . .

> ⚠ **Warning**
>
> But you should not simply use them to replace your learning.

## How to learn programming

### Our Recommendation

1. Be present: Attend the lecture and solve the tutorials
2. Put in some work: Repeat code and try to understand it
3. Do coding: Run code, play around, modify, and solve

. . .

> 💡 **Tip**
>
> Great resources to start are books and small challenges. You can find a list of book recommendations at the end of the lecture. Small challenges to solve can for example be found on Codewars.

### Don't give up!

- Programming is problem solving, don't get frustrated!
- Expect to stretch your comfort zone

## Setting up Python

### Install VS Code

- Download and install from the website
- Built for Windows, Linux and Mac
- Install the Python and Jupyter extension
- Now you are ready to go!

. . .

> 💡 Tip
>
> Unsure on how to work with VS Code and notebooks? Ask us! We are happy to help you out!

## What is an IDE?

- Integrated Development Environment = application
- It allows you to write, run and debug code scripts
- Other IDEs include for example:
  ‣ [PyCharm](#) from JetBrains
  ‣ [Zed](#)

## Installation of Python with `uv`

- We will use `uv` to install and manage Python versions
- It works on Windows, Mac and Linux
- It helps us to manage packages and virtual environments
- Now, we all [go here](#) and install `uv` and Python

. . .

> 💡 Tip
>
> If the installation does not work, let us know!

# Using Notebooks with uv

## Quick Check

- Have you installed `uv` and initialized the project?
- Great! Before we continue, check the following:
  ‣ [] You have a folder for the course
  ‣ [] You have initialized `uv` with `uv init` inside the folder
  ‣ [] You can see a file called `pyproject.toml` in the folder

. . .

> 💡 Tip
>
> Something not working yet? Ask us!

## Using Notebooks

- Now we need to add a kernel to our project
- Run `uv add --dev ipykernel` from your terminal
- Now run `uv add jupyter` in the terminal
- This allows us to use `uv` Python in notebooks

- Done? Perfect. Now we can start!

## Working with Notebooks
- Now you can download the files from the website
- Just click on one of the sessions and open it
- Select `Jupyter` on the right side
- Download and save the files to your course folder
- Open them and select "Open with Jupyter Notebook"

. . .

> 💡 Tip
>
> That was the hardest part today!

Any questions

so far?

## After the break — Optimal Stopping
- The "Secretary Problem" and the 37% rule
- When to stop searching and make a decision
- How to translate the idea into code and experiments

> ℹ Note
>
> That's it for our introduction!
> Let's have a short break and then continue with our first topic.

# Literature

## Interesting literature to start
- Christian, B., & Griffiths, T. (2016). Algorithms to live by: the computer science of human decisions. First international edition. New York, Henry Holt and Company.[2]
- Ferguson, T.S. (1989) 'Who solved the secretary problem?', Statistical Science, 4(3). doi:10.1214/ss/1177012493.

## Books on Programming
- Downey, A. B. (2024). Think Python: How to think like a computer scientist (Third edition). O'Reilly. [Here](Here)
- Elter, S. (2021). Schrödinger programmiert Python: Das etwas andere Fachbuch (1. Auflage). Rheinwerk Verlag.

. . .

---

[2]The main inspiration for this lecture. Nils and I have read it and discussed it in depth, always wanting to translate it into a course.

> **ⓘ Note**
>
> Think Python is a great book to start with. It's available online for free. Schrödinger Programmiert Python is a great alternative for German students, as it is a very playful introduction to programming with lots of examples.

## More Literature

For more interesting literature, take a look at the [literature list](#) of this course.