Introduction to Computing for the Social Sciences

WS 2025/26
Lecturer: David Garcia
Tutors: Anastasia Siebers, Niklas Bacher

Lecture: Tuesdays 10:00-11:30, P1138
Exercises: Thursdays 13:30-15:00, K503
Tutorial 1: Mondays 15:15 – 16:45, C422 Tutorial 2: Wednesdays 15:15-16:45, P812

Course Description

This introductory course on computation is designed to expose social science students with no background in computing to the fundamental concepts of computer science, basics of programming, and their role in the social sciences and society. We will be covering a gamut of topics starting from a basic overview of computer systems, information coding, and storage, to data structures, algorithms, and database systems. The course has a special emphasis on a social informatics perspective to illustrate the applications of computing in the social sciences, the role of computer science in society, and how social and historical forces have shaped informatics. Theoretical lectures will be accompanied by practical programming exercises (in Python programming language) that will allow students to directly apply the acquired knowledge to application scenarios relevant to the social sciences. **The main goal of the course is to equip students with sufficient basic conceptual understanding of computer science and programming to be able to solve problems computationally.**

Pre-requisites

This is a starter-level course but assumes some basic previous experience with programming in Python. In preparation for the course, we recommend attending the Python Block Course in October.

Learning objectives

Upon completion of this course, students will be able to:

- Understand and use ideas at the core of computer science and programming
- Demonstrate familiarity with major algorithms and data structures
- Use computational techniques for solving problems in the social sciences
- Have familiarity with Python programming language for simple computational tasks

The course has three types of sessions:

Lectures
The course has a weekly, mandatory 90-minute lecture that introduces concepts and foundations in computer science. The slides for these lectures are made available on Ilias the day before the lecture takes place.

Exercises
The course includes optional extra 90-minute sessions to discuss both pen-and-paper and practical coding exercises. These are aligned with the lectures and help to bridge theory and practice. Exercise sheets and notebooks will be distributed over Ilias in advance and will serve as the basis for these sessions. These exercises help both as practice for the final written exam and to develop skills for the final projects.

Tutorials
Tutorials help students to apply fundamental concepts of computer science in Python. Tutorial sessions serve as contact points for students to develop practical skills and to work and discuss solutions on programming assignments in the Python programming language. There will be two tutorial groups covering the same content each week. You don't have to register to a group and we will only ask for registration if attendance is very imbalanced.

How to get help from us during this course:
- Ask questions on the course's github forum: https://github.com/dgarcia-eu/ICSS/discussions/
- During lectures, exercise sessions and tutorials
- Book a consultation hour with David Garcia on Ilias

Course literature
Given the breadth of topics covered in the course, it is difficult to recommend a single book for the course. We provide specific references for each topic in the course outline. The following books cover some of the contents of the course:

On fundamentals and theory of computer science:
- 'Introduction to Computing - Explorations in Language, Logic, and Machines' by Evans https://computingbook.org/ (Creative Commons, 2019 version)
- 'Introduction to the Theory of Computation' by Sipser (2012, Third edition)
- 'Python Programming: An introduction to Computer Science' by Zelle (2016, Third edition)
- Introduction to Computation and Programming Using Python' by Guttag (2016, Second edition)

On algorithms and data structures:
- 'Algorithms Unlocked' by Cormen (2013)
- 'Introduction to Algorithms' by Cormen, Leiserson, Rivest, and Stein (2009, Third edition)

Grading

The course is graded in two parts:

Introduction to Computing for the Social Sciences (Theory) | POL-31760 (6 ECTS)

The grade for the theory part is based on a final 90-minute written exam. Each year, there will be a resit exam for those who fail or do not take the first date. If you pass in the first date, you cannot go to the resit exam to improve your grade.

Introduction to Computing for the Social Sciences (Practice) | POL-31770 (6 ECTS)

1 - Mandatory assignments. The practical part has four mandatory programming assignments during the semester. To be allowed to submit a final project for the practice part, a student has to **pass each of the four assignments,** i.e. to get at least 50% of the points in each assignment.

2 - Final practical project. The grade of the practical part is based on a final project. To be allowed to submit this project, you have to pass each of the four mandatory assignments. In this final project, individual students showcase their practical learning through a programming project of their own choice. The submission of the final project is composed of the project code, a report, and a short video explaining the code. More details about the video format, submission, and examples will be posted and presented in the course before the holiday break. The deadline for the final projects will be close to the end of the semester (last weeks of March).

Note on grading: Each part (theory and practice) is graded separately and it is possible to pass one and not the other. For the theory part there is a resit exam and the standard number of tries is two. You do not need to pass both parts the same year but we strongly recommend that you focus on passing both on your first semester to be ready for CS courses from the second semester.

Honor code: We expect students to adhere to an honor code. We do encourage students to discuss the topics taught in the course with each other and to take the help of online resources for learning more about these topics. However, we expect your answers to be your own and for students to submit code written or programmed by themselves only. **It is not allowed to copy code from online resources for your submissions, including AI coding assistants.** You will have to submit an authorship statement with your final project and abuse of AI coding assistants can lead to plagiarism investigations.

Course Schedule

Week 1 (20.10 - 24.10)

- Lecture: Introduction
  What is computer science; what is a computer; what is a program.
  Reference(s): Chapter 1 of 'Python Programming: An introduction to Computer Science' by Zelle
- Exercise: Introduction to Github
- Tutorial: Console & Virtual Environments

Week 2 (27.10 - 31.10)

- Lecture: Information Coding
  Units of information - bits; Representing different types of data using bits.
  Reference(s): Chapter 1 of 'Introduction to Computing - Explorations in Language, Logic, and Machines' by Evans
- Exercise: Information coding and binary operations
- Tutorial: Python for ICSS

Week 3 (3.11 - 7.11)

- Lecture: Data Structures
  Abstract data types: Linked lists; Stacks; Queues; Trees; Hash tables.
  Reference(s): Chapter 10 of 'Introduction to Algorithms' by Cormen, Leiserson, Rivest and Stein
- Exercise: Data structure exercises
- Tutorial: Scraping

Deadline for assignment 1: Sunday, 09.11, 23:59

Week 4 (10.11 - 14.11)

- Lecture: Programming
  Programming languages; Operators; Decisions; Procedures; Structured programming
  Reference(s): Chapters 2, 6-8 of 'Python Programming: An introduction to Computer Science' by Zelle
- Exercise: Debugging and code quality
- Tutorial: APIs

Week 5 (17.11 - 21.11)

- Lecture: Algorithms
  What is an algorithm; Search algorithms.
  Reference(s): Chapter 1 of 'Algorithms Unlocked' by Cormen
- Exercise: Advanced Git
- Tutorial: Networks

Week 6 (24.11 - 28.11)

- Lecture: Recursion
  Recursive algorithms; Recursion vs. iteration; Properties of recursive algorithms
  Reference(s): Chapter 13 of 'Python Programming: An introduction to Computer Science' by Zelle
- Exercise: Graph algorithms
- Tutorial: Streamlit

Deadline for assignment 2: Sunday, 30.11, 23:59

Week 7 (1.12 - 5.12)

- Lecture: Sorting Algorithms
  Selection sort; Insertion sort; Divide and conquer algorithms; Merge sort; Quick sort.
  Reference(s): Chapter 2, 3 of 'Algorithms Unlocked' by Cormen.
- Exercise: Sorting Algorithms
- Tutorial: NLP and exception handling

Week 8 (8.12 - 12.12)

- Lecture: Time Complexity
  Runtime analysis; Asymptotic notation; Applications to search and sorting
  Reference(s): Chapter 9 & 10 of 'Introduction to Computation and Programming Using Python' by Guttag
- Exercise: Regular expressions
- Tutorial: Parallelism and runtime analysis

Week 9 (15.12 - 19.12)

- Lecture: Theoretical Computer Science
  Models of computation (DFA, PDA, Turing Machines), Complexity Classes
  Reference(s): Chapter 1 of 'Introduction to the Theory of Computation' by Sipser. Chapters 6 & 12 of 'Introduction to Computing - Explorations in Language, Logic, and Machines' by Evans. Chapter 10 of 'Algorithms Unlocked' by Cormen
- Exercise: Algorithmic complexity
- Tutorial: Q&A, assignment support

Deadline for assignment 3: Sunday, 21.12, 23:59

Week 10 (5.1 - 9.1)

- Lecture: Machine Learning
  Supervised and unsupervised machine learning. Bias-variance tradeoff
  Reference(s): Lecture materials
- Exercise: Recap on Theoretical Computer Science
- Tutorial: Virtual Machines

Week 11 (12.1 - 16.1)

- Lecture: Databases
  Database management systems; Tables, SQL, relational algebra
  Reference(s): Chapter 4 of 'Big Data and Social Science: A Practical Guide to Methods and Tools' by Foster, Ghani, Jarmin, Kreuter and Lane.
- Exercise: Database design and SQL
- Tutorial: Machine Learning in Python

Week 12 (19.1 - 23.1)

- Lecture: Parallel Programming
  Hardware and software parallelization, limits of parallelization
  Reference(s): Chapter 4.8 of 'Composing Programs' by DeNero.
  Chapter 5 of 'Big Data and Social Science: A Practical Guide to Methods and Tools' by Foster, Ghani, Jarmin, Kreuter and Lane.
- Exercise: JQ
- Tutorial: Relational Databases

Deadline for assignment 4: Sunday, 25.1, 23:59

Week 13 (26.1 - 30.1)

- Lecture: Social Science Applications
  Applications of computational approaches to social science research including measurement, analysis, theory-building, and data retrieval.
  Reference(s): Lecture materials
- Exercise: Practical projects Q&A and feedback
- Tutorial: Q&A

Week 14 (2.2 - 6.2)

- Lecture: Exam Review
  Recap on the rest of the course
- Exercise: Practical projects Q&A and feedback
- Tutorial: Q&A