

Algorithms and Data Structures

A1. Organizational Matters

Gabriele Röger and Patrick Schneider

University of Basel

February 19, 2025

Organizational Matters

People



Gabriele Röger



Patrick Schnider

Lecturer

Gabi Röger

- **email:** `gabriele.roeger@unibas.ch`
- **office:** room 04.005, Spiegelgasse 1

Patrick Schnider

- **email:** `patrick.schnider@unibas.ch`
- **office:** room 04.009, Spiegelgasse 1

People



Tutors

Floyd Peiszan (floyd.peiszan@stud.unibas.ch)

- Tuesday, 14.15-16.00, Pharmazentrum, U1075

Maya Posva (maya.posva@stud.unibas.ch)

- Wednesday, 10.15-12.00, Pharmazentrum, U1075

Yanick Spichty (y.spichty@stud.unibas.ch)

- Friday, 14.15-16.00, Pharmazentrum, U1075

Time & Place

Lectures

- **Wednesday:** 14:15–16:00, Biozentrum, lecture hall U1.131
- **Thursday:** 14:15–16:00, Alte Universität, lecture hall -101

Exercise Sessions (starting February 21/25/26)

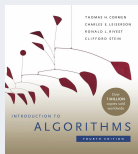
- **Tuesday,** 14.15-16.00, Pharmazentrum, U1075
- **Wednesday,** 10.15-12.00, Pharmazentrum, U1075
- **Friday,** 14.15-16.00, Pharmazentrum, U1075

Resources

- **Adam:** central starting point and exercises
<https://adam.unibas.ch/>
- **Website:** course information, slides, notebooks
- **Discord:** for your interaction with each other
 - Idea: course participants help each other.
 - Lecturers and tutors can help by request.
 - Feel free to use a **pseudonym**.

Textbook

Textbook



Introduction to Algorithms
by Thomas H. Cormen, Charles E. Leiserson,
Ronald L. Rivest and Clifford Stein
(MIT Press, Fourth Edition)

Programming Languages

- Lectures: Mostly Python
→ Advantage: compact and direct, ideal for smaller programs
- Exercises: Java or Python (indicated on exercise sheet)



We don't require any previous knowledge about Python!

Exercises

Exercise sheets (homework assignments):

- theoretical and programming exercises
- on ADAM every Thursday evening
- may be solved in groups (we recommend groups of 2-3)
- group members should be in same exercise group
- due Friday the following week (23:55)
(upload to Adam at <https://adam.unibas.ch/>)
- discussion and **individual feedback** in exercise meeting

Exercises

Exercise sessions:

- introduction of/questions about the current exercise sheet
- discussion of previous exercise sheet (common problems)
- questions about the course
- if time: work on the homework assignment
 - support with the current exercise sheet
 - technical support (Java/Python, programming environment)
- participation voluntary but highly recommended

Exercises

Exercise sessions:

- introduction of/questions about the current exercise sheet
- discussion of previous exercise sheet (common problems)
- questions about the course
- if time: work on the homework assignment
 - support with the current exercise sheet
 - technical support (Java/Python, programming environment)
- participation voluntary but highly recommended

important: please fill in the survey on ADAM for the group assignment until **tomorrow 15:15** (February 20).

- One registration per team (please list all names).
- All team members will be in the same exercise session.

Course Format

- 6 ECTS main course + 2 ECTS exercises
- separate enrolment and evaluation
- can and should be taken in parallel

Enrolment

- <https://services.unibas.ch/>
- register today for the course, so that you get all relevant emails and access to the ADAM workspace
- enrolment for exercise after we made the group assignment

Prerequisites

- basic programming skills (ideally Java or Python)

Evaluation of Main Course (6 CP)

- **written exam**, 6 ECTS credits, graded 1-6
- 11 June 2025, 14:00-16:00, Biozentrum, Hörsaal U1.111, Maurice E. Müller Saal
- admission to exam: **no prerequisites**
- must **register** for exam during March 31 – April 14
↪ see <https://philnat.unibas.ch/de/examen/>
- grade for course determined exclusively by the exam
- if you fail: **one** repeat attempt (within one year)

Evaluation of Main Course (6 CP)

- **written exam**, 6 ECTS credits, graded 1-6
- 11 June 2025, 14:00-16:00, Biozentrum, Hörsaal U1.111, Maurice E. Müller Saal
- admission to exam: **no prerequisites**
- must **register** for exam during March 31 – April 14
↪ see <https://philnat.unibas.ch/de/examen/>
- grade for course determined exclusively by the exam
- if you fail: **one** repeat attempt (within one year)

Last lecture (May 28): Q&A session for exam preparation

Evaluation of Exercises (2 CP)

- midterm exams on April 2 and May 8
- in the usual lecture hall (Biozentrum)
- pass/fail evaluation based on the accumulated marks from the midterm exams

Laptops

Small exercises during the lecture: please bring your laptop.

But stay focused:

Research Article

Logged In and Zoned Out: How Laptop Internet Use Relates to Classroom Learning

Susan M. Ravizza, Mitchell G. Uitvlugt, and Kimberly M. Fenn

Department of Psychology, Michigan State University, East Lansing

Abstract

Laptop computers are widely prevalent in university classrooms. Although laptops are a valuable tool, they offer access to a distracting temptation: the Internet. In the study reported here, we assessed the relationship between classroom performance and actual Internet usage for academic and nonacademic purposes. Students who were enrolled in an introductory psychology course logged into a proxy server that monitored their online activity during class. Past

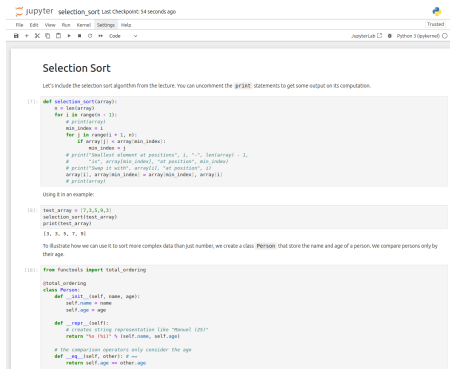


Psychological Science
2017, Vol. 28(2) 171–180
© The Author(s) 2016
Reprints and permissions:
sagepub.com/journalsPermissions.nav
DOI: 10.1177/09567976166677314
www.psychologicalscience.org/PS

Jupyter Notebooks

web-based interactive computational environment for Python
(and some other languages)

- illustrating algorithms and concepts
- implementing algorithms for experimenting and studying at home
- small exercises during the lecture



The screenshot shows a Jupyter Notebook titled "selection_sort" with a toolbar and a "Trust" button. The notebook content includes:

Selection Sort

Let's include the selection sort algorithm from the lecture. You can uncomment the `#PREREQ` statements to get some output on its computation.

```
171 def selection_sort(array):
172     n = len(array)
173     for i in range(n - 1):
174         # print(array)
175         min_index = i
176         for j in range(i + 1, n):
177             # PREREQ: array[min_index]
178             min_index = j
179             # print("Smallest element at positions", i, "-", j, len(array) - 1,
180                   # "is", array[min_index], "at position", min_index)
181             # print("Swap if with", array[i], "at position", i)
182             array[i], array[min_index] = array[min_index], array[i]
183         # print(array)
```

Using it in an example:

```
184 test_array = [7, 2, 5, 9, 21]
185 selection_sort(test_array)
186 print(test_array)
187 [2, 5, 7, 9, 21]
```

To illustrate how we can use it to sort more complex data than just numbers, we create a class `Person` that store the name and age of a person. We compare persons only by their age.

```
188 from functools import total_ordering
189 @total_ordering
190 class Person:
191     def __init__(self, name, age):
192         self.name = name
193         self.age = age
194
195     def __repr__(self):
196         # created string representation like "Maxwell (28)"
197         return "%s (%i)" % (self.name, self.age)
198
199     # the comparison operators only consider the age
200     def __eq__(self, other):
201         return self.age == other.age
```

Questions on Organization



Questions?

About this Course

Algorithms and Data Structures

- some basic building blocks are needed again and again in programming projects, e.g.
 - sorting algorithms
 - search trees
 - priority queues
 - shortest path in a graph
 - ...
- often provided by libraries

Algorithms and Data Structures

- some basic building blocks are needed again and again in programming projects, e.g.
 - sorting algorithms
 - search trees
 - priority queues
 - shortest path in a graph
 - ...
- often provided by libraries
- here you learn ...
 - how all this works internally.
 - how to select suitable building blocks.
 - tricks to achieve efficient programs.

Algorithms and Data Structures

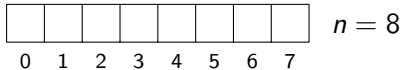
- some basic building blocks are needed again and again in programming projects, e.g.
 - sorting algorithms
 - search trees
 - priority queues
 - shortest path in a graph
 - ...
- often provided by libraries
- here you learn ...
 - how all this works internally.
 - how to select suitable building blocks.
 - tricks to achieve efficient programs.
- independent of specific programming language

Example: Algorithms for Sorting

- task: sort a sequence of elements in increasing order, e.g.
input [5, 9, 3, 5] → result [3, 5, 5, 9]
- 1960s (and a long time afterwards): a quarter of all commercial computation time used for sorting
- naive algorithm: **selection sort**



Selection Sort: Informally



- identify smallest element at positions $0, \dots, n - 1$ and swap it to position 0
- identify smallest element at positions $1, \dots, n - 1$ and swap it to position 1
- ...
- identify smallest element at positions $n - 2, n - 1$ and swap it to position $n - 2$

Selection Sort: Example

3	7	2	9	7	1	4	5
---	---	---	---	---	---	---	---

Selection Sort: Example

3	7	2	9	7	1	4	5
---	---	---	---	---	---	---	---

1	7	2	9	7	3	4	5
---	---	---	---	---	---	---	---

Selection Sort: Example

3	7	2	9	7	1	4	5
---	---	---	---	---	---	---	---

1	7	2	9	7	3	4	5
---	---	---	---	---	---	---	---

1	2	7	9	7	3	4	5
---	---	---	---	---	---	---	---

Selection Sort: Example

3	7	2	9	7	1	4	5
---	---	---	---	---	---	---	---

1	7	2	9	7	3	4	5
---	---	---	---	---	---	---	---

1	2	7	9	7	3	4	5
---	---	---	---	---	---	---	---

1	2	3	9	7	7	4	5
---	---	---	---	---	---	---	---

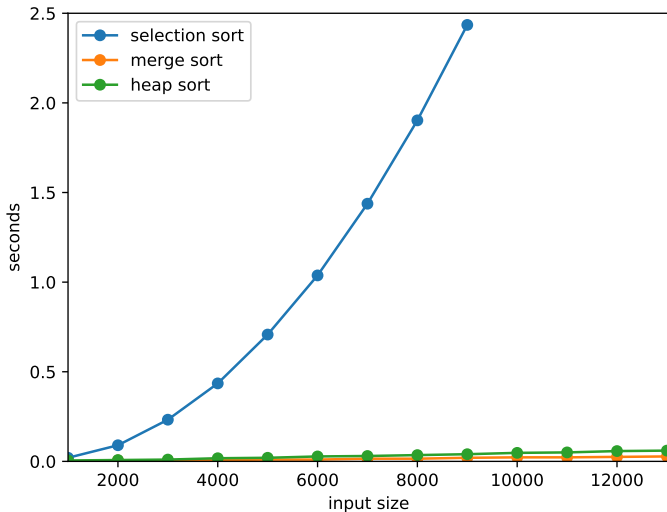
1	2	3	4	7	7	9	5
---	---	---	---	---	---	---	---

1	2	3	4	5	7	9	7
---	---	---	---	---	---	---	---

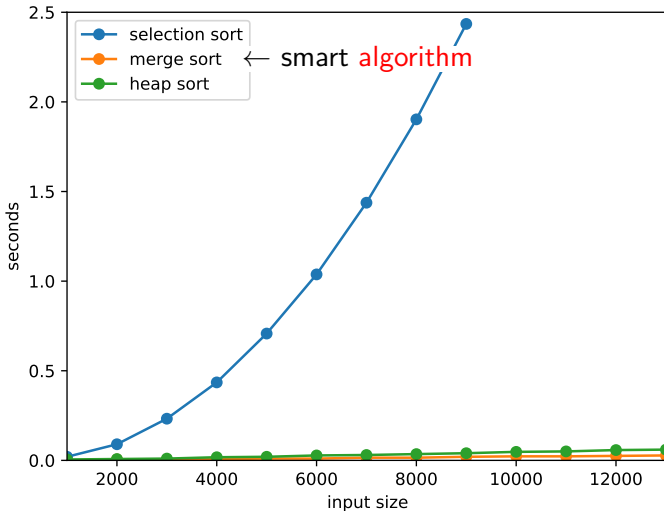
1	2	3	4	5	7	9	7
---	---	---	---	---	---	---	---

1	2	3	4	5	7	7	9
---	---	---	---	---	---	---	---

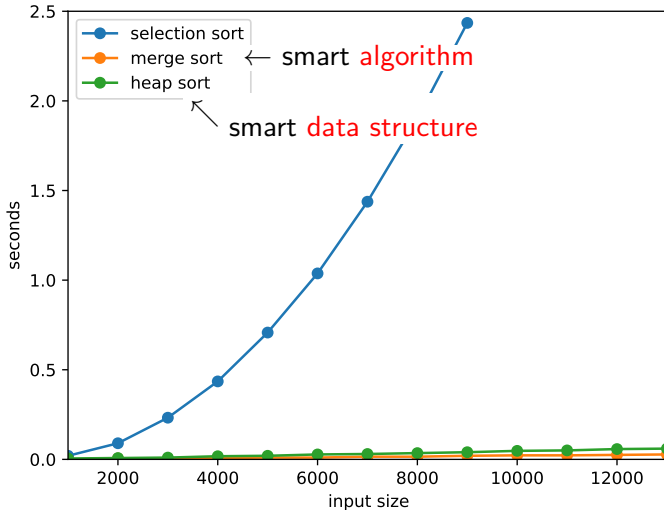
Algorithms for Sorting: Runtime



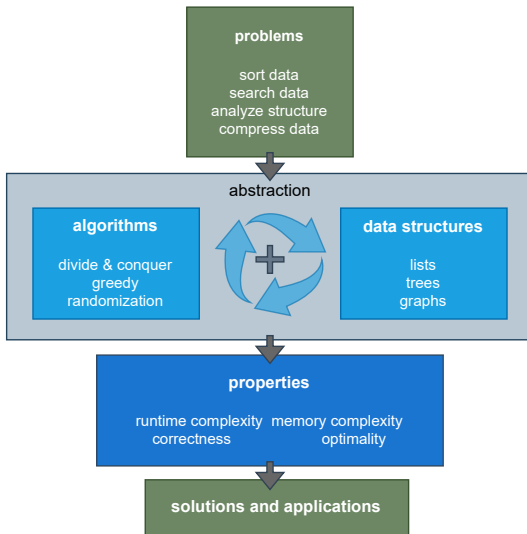
Algorithms for Sorting: Runtime



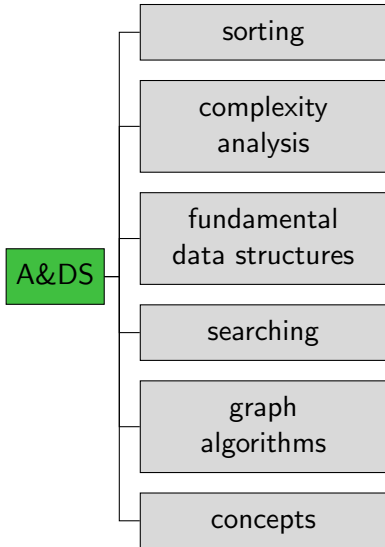
Algorithms for Sorting: Runtime



Algorithms and Data Structures



Content of the Course



Content of the Course

