Algorithmen und Datenstrukturen

A1. Organisatorisches

Marcel Lüthi and Gabriele Röger

Universität Basel

20. Februar 2019

A1.1 Organisatorisches

A1.2 Über diese Vorlesung

A1.1 Organisatorisches

Personen: Dozenten

Marcel Lüthi

Gabriele Röger

Dozenten

Dr. Marcel Lüthi

- E-Mail: `marcel.luethi@unibas.ch`
- Büro: Raum 04.002, Spiegelgasse 1

Dr. Gabriele Röger

- E-Mail: `gabriele.roeger@unibas.ch`
- Büro: Raum 04.005, Spiegelgasse 1

Personen: Tutoren

Tutoren

Clemens Büchner
- E-Mail: `clemens.buechner@unibas.ch`
- Mittwoch, 10-12 Uhr

Simon Peterhans
- E-Mail: `simon.peterhans@unibas.ch`
- Freitag 14-16 Uhr

Lukas Stöckli
- E-Mail: `lukas.stoeckli@unibas.ch`
- Dienstag 14-16 Uhr

Zeit & Ort

Vorlesungen
- Mi 14:15-16:00 Uhr, Kollegienhaus Hörsaal 120
- Do 14:15-16:00 Uhr, Kollegienhaus Hörsaal 116

Übungen
- Spiegelgasse 1, Computer-Labor U1.001
- Di 14:15-16:00
- Mi 10:15-12:00
- Fr 14:15-16:00

Erster Übungstermin 22./26./27. Februar

Vorlesung im Web

Vorlesungsseite

`https://dmi.unibas.ch/de/studium/computer-science-informatik/lehrangebot-fs19/vorlesung-algorithmen-und-datenstrukturen/`
- Vorlesungsbeschreibung
- Folien
- Zusatzmaterial (nicht prüfungsrelevant)

Anmeldung:
- `https://services.unibas.ch/`
- Bitte registrieren Sie sich gleich heute, um alle kursrelevanten Informationen zu erhalten.
- Bitte tragen Sie sich auch für eine Übungsgruppe ein (unter `https://courses.cs.unibas.ch`).

Vorlesungsmaterialien

Vorlesungsmaterialien:
- Vorlesungsfolien (online)
- Lehrbuch
- vertiefendes Material auf Anfrage

Lehrbuch

Algorithmen
von Robert Sedgewick und Kevin Wayne
(Pearson Verlag, 4. Auflage)

Weitere nützliche Ressourcen

- Seite zum Buch: Algorithms, 4th edition:
  `https://algs4.cs.princeton.edu/home/`
- Youtube (Suchbegriff: Robert Sedgewick algorithms)
- Data Structures and Algorithms – The Basic Toolbox
  von Kurt Mehlhorn und Peter Sanders (Springer Verlag)
  `http://people.mpi-inf.mpg.de/~mehlhorn/Toolbox.html`
- Google, Wikipedia, ...

Zielgruppe

Zielgruppe:

- Bachelor Informatik (ab 2. Semester)
- Bachelor Computational Sciences (ab 2. Semester)
- Alle Studierenden mit Programmierkenntnissen
  sind herzlich willkommen.

Voraussetzungen:

- Programmierung (Java)

Programmiersprachen

- Vorlesung: Hauptsächlich Python
  → Vorteil: Kompakt und direkt, ideal für kleine Programme
- Übungen: Java oder Python (nach Ankündigung)

Es werden keine Python-Kenntnisse vorausgesetzt!

Übungen

Übungsaufgaben:

- Hausaufgaben (Theorie + Praxis)

Übungstermine:

- Besprechung der Hausaufgaben
- Beantwortung von Fragen zum aktuellen Blatt
- Technische Hilfestellung (Java/Python,
  Programmierumgebung)
- Teilnahme freiwillig - aber sehr empfohlen.

Übungen: Hausaufgaben

Hausaufgaben:

- Aufgaben ab Donnerstagabend auf Adam verfügbar.
- Bearbeitung allein oder in Zweiergruppen ($2 \neq 3$)
- Abgabe freitags in Folgewoche (23:59) auf Adam
- Besprechung und individuelles Feedback in Übungsgruppe

Prüfung

- schriftliche Prüfung
- Mo, 17. Juni 2019, 15-17 Uhr
- 8 Kreditpunkte
- Zulassung:
  - Alle bis auf höchstens zwei Übungsblätter (9 von 11) erfolgreich bearbeitet
  - „erfolgreich bearbeitet“ = mind. 60% der Punkte
- Note basiert nur auf Klausur
- keine Wiederholungsprüfung

Plagiate

Plagiat (Wikipedia)

> *Ein Plagiat ([...] aus lateinisch plagiarius, deutsch „Seelenverkäufer, Menschenräuber“) ist die Anmaßung fremder geistiger Leistungen. Dies kann sich auf die Übernahme fremder Texte oder anderer Darstellungen [...], fremder Ideen [...] oder beides gleichzeitig [...] beziehen.*

Folge:

- 0 Punkte für Übungsblatt (beim ersten Mal)
- Nicht-Zulassung zur Prüfung (im Wiederholungsfall)

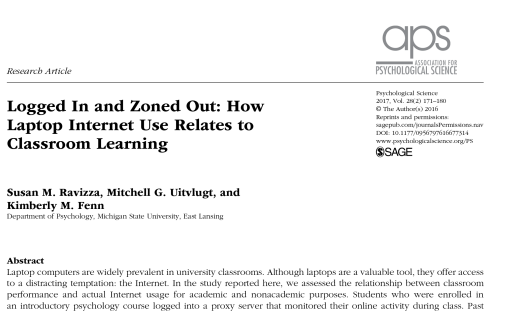
Im Zweifelsfall: vorher klären, was (nicht) in Ordnung ist

Aufgaben zu schwer? Wir helfen gerne!

Laptops

Miniübungen während der Vorlesung: Bitte Laptops mitbringen.

Aber Vorsicht:



Research Article

aps ASSOCIATION FOR PSYCHOLOGICAL SCIENCE

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

Psychological Science
2017, Vol. 28(2) 171–180
© The Author(s) 2016
Reprints and permissions:
sagepub.com/journalsPermissions.nav
DOI: 10.1177/0956797616677314
www.psychologicalscience.org/PS
SAGE

Susan M. Ravizza, Mitchell G. Uitvlugt, and Kimberly M. Fenn
Department of Psychology, Michigan State University, East Lansing

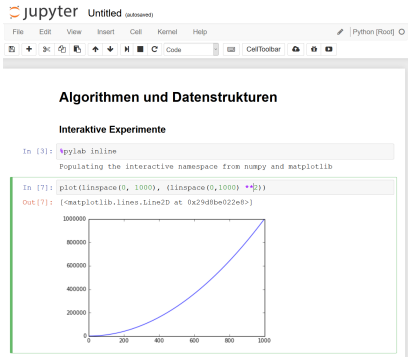
Abstract
Laptop computers are widely prevalent in university classrooms. Although laptops are a valuable tool, they offer access to a distracting temptation: the Internet. In the study reported here, we assessed the relationship between classroom performance and actual Internet usage for academic and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

Jupyter-Notebooks

Web-basierte interaktive Arbeitsumgebung für Python

Nutzung von Jupyter-Notebooks:

- Erklärungen zu Algorithmen
- Implementation von Algorithmen
  - Ideal zum Experimentieren und Lernen
- Miniübungen während der Vorlesung



Jupyter-Notebooks – Online version

- Alle Jupyter Notebooks könnnen online aufgerufen und bearbeitet werden
  - Link jeweils auf Vorlesungsseite
- Vorteile:
  - Keine Installation erforderlich
  - Immer up-to-date
- Nachteile:
  - Änderungen werden nicht gespeichert

Jupyter-Notebooks – Lokale Installation

- Lokale Installation (via Anaconda)
- Vorteile:
  - Änderungen werden lokal gespeichert
  - Schneller
- Nachteil:
  - Notebooks müssen synchronisiert werden (mit git oder manuell)

- Github repository:
  `https://github.com/marcelluethi/algodata-jupyter-notebooks`
- Installationsanleitung
  `https://jupyter.readthedocs.io/en/latest/install.html`

Fragen zur Organisation

Fragen?

A1.2 Über diese Vorlesung

Algorithmen und Datenstrukturen

- Bestimmte Grundbausteine benötigt man immer wieder bei Programmierprojekten, z.B.
  - Sortierverfahren
  - Suchbäume
  - Prioritätswarteschlangen
  - kürzeste Pfade in Graphen
  - ...
- Wird oftmals durch Bibliotheken fertig bereitgestellt.
- Hier lernen Sie ...
  - wie das alles intern funktioniert.
  - wie man den richtigen Baustein auswählt.
  - Tricks und Kniffe für effiziente Programme.
- Methoden unabhängig von konkreter Programmiersprache

Beispiel: Sortieralgorithmen

- Aufgabe: Bringe Sequenz von Elementen in aufsteigende Reihenfolge, z.B. Eingabe [5, 9, 3, 5] $\rightarrow$ Ausgabe [3, 5, 5, 9]
- 1960er Jahre (und noch lange danach): ein Viertel der kommerziell verbrauchten Rechenzeit für Sortiervorgänge
- Naiver Algorithmus: Selectionsort

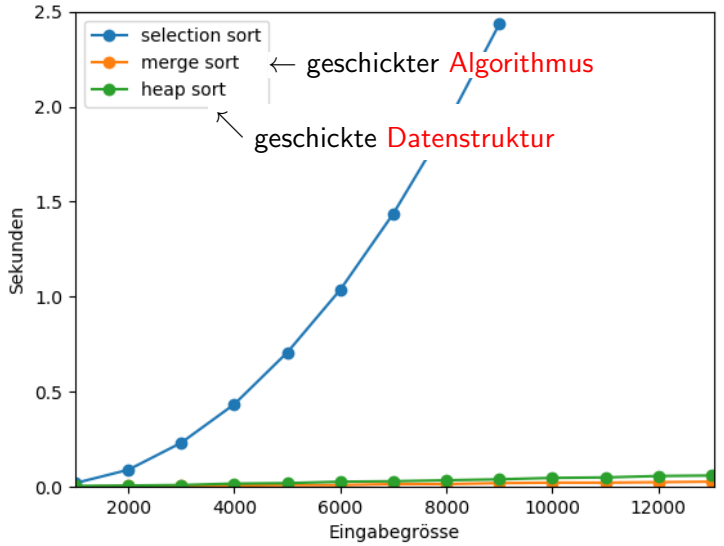
Selectionsort: Informell

- Finde kleinstes Element an Positionen $0, \ldots, n-1$ und tausche es an Position 0
- Finde kleinstes Element an Positionen $1, \ldots, n-1$ und tausche es an Position 1
- ...
- Finde kleinstes Element an Positionen $n-2, \ldots, n-1$ und tausche es an Position n - 2

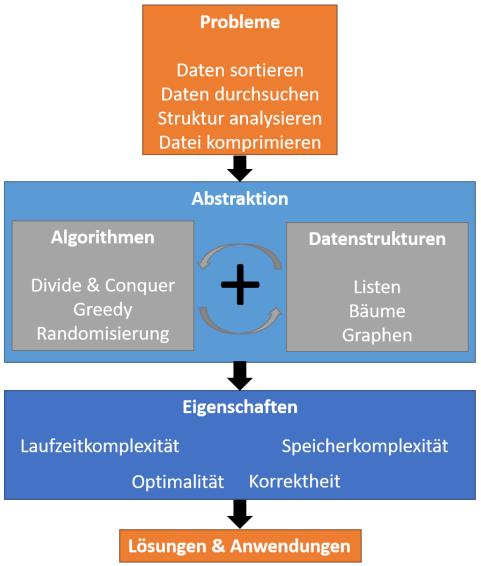
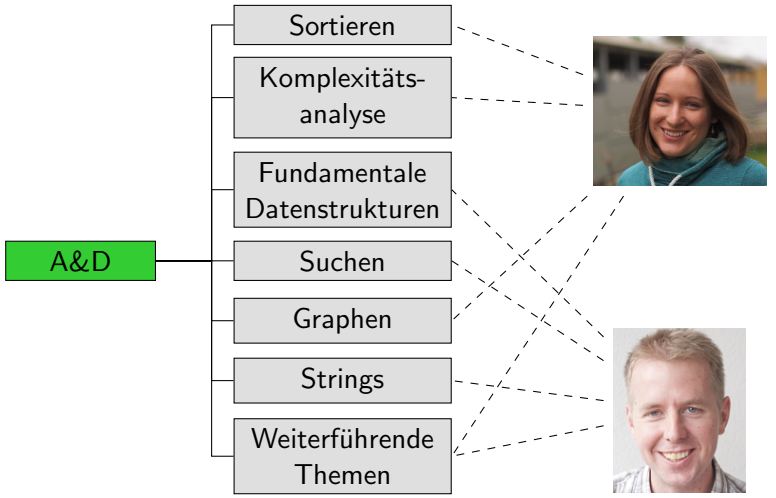
Selectionsort: Beispiel

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

Sortieralgorithmen: Laufzeit



Der Kurs Algorithmen und Datenstrukturen



Inhalt dieser Veranstaltung