Programmierung und Deskriptive Statistik

BSc Psychologie WiSe 2025/26

Belinda Fleischmann und Dirk Ostwald

	Gruppe 1/2	Gruppe 3	Format	Thema
1	**Mi, 15.10.**	**Do, 16.10.**	**Seminar**	**(1) Grundbegriffe der Informatik**
2	Mi, 22.10.	Do, 23.10.	Seminar	(2) Arithmetik und Variablen
3	Mi, 29.10.	Do, 30.10.	Übung	(2) Arithmetik und Variablen
4	Mi, 05.11.	Do, 06.11.	Seminar	(3) Vektoren und Matrizen
5	Mi, 12.11.	Do, 13.11.	Übung	(3) Vektoren und Matrizen
6	Mi, 19.11.	Do, 20.11.	Seminar	(4) Listen und Dataframes
7	Mi, 26.11.	Do, 27.11.	Übung	(4) Listen und Dataframes
8	Mi, 03.12.	Do, 04.12.	Seminar	(5) Datenmanagement
9	Mi, 10.12.	Do, 11.12.	Übung	(5) Datenmanagement
	Mo, 15.12		Abgabe	*Individualleistung (1/2)*
10	Mi, 17.12.	Do, 18.12.	Seminar	(6) Strukturiertes Progr.: Kontrollfluss, Debugging
11	Mi, 07.01.	Do, 08.01.	Seminar	(7) Häufigkeitsverteilungen
12	Mi, 14.01.	Do, 15.01.	Übung	(7) Häufigkeitsverteilungen
13	Mi, 21.01.	Do, 22.01.	Seminar	(8) Maße der zentralen Tendenz und Datenvariabilität
14	Mi, 28.01.	Do, 29.01.	Übung	(8) Maße der zentralen Tendenz und Datenvariabilität
	Mo, 02.02		Abgabe	*Individualleistung (2/2)*

(1) Grundbegriffe der Informatik

Datenanalyse

Informatik

Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Datenanalyse

Informatik

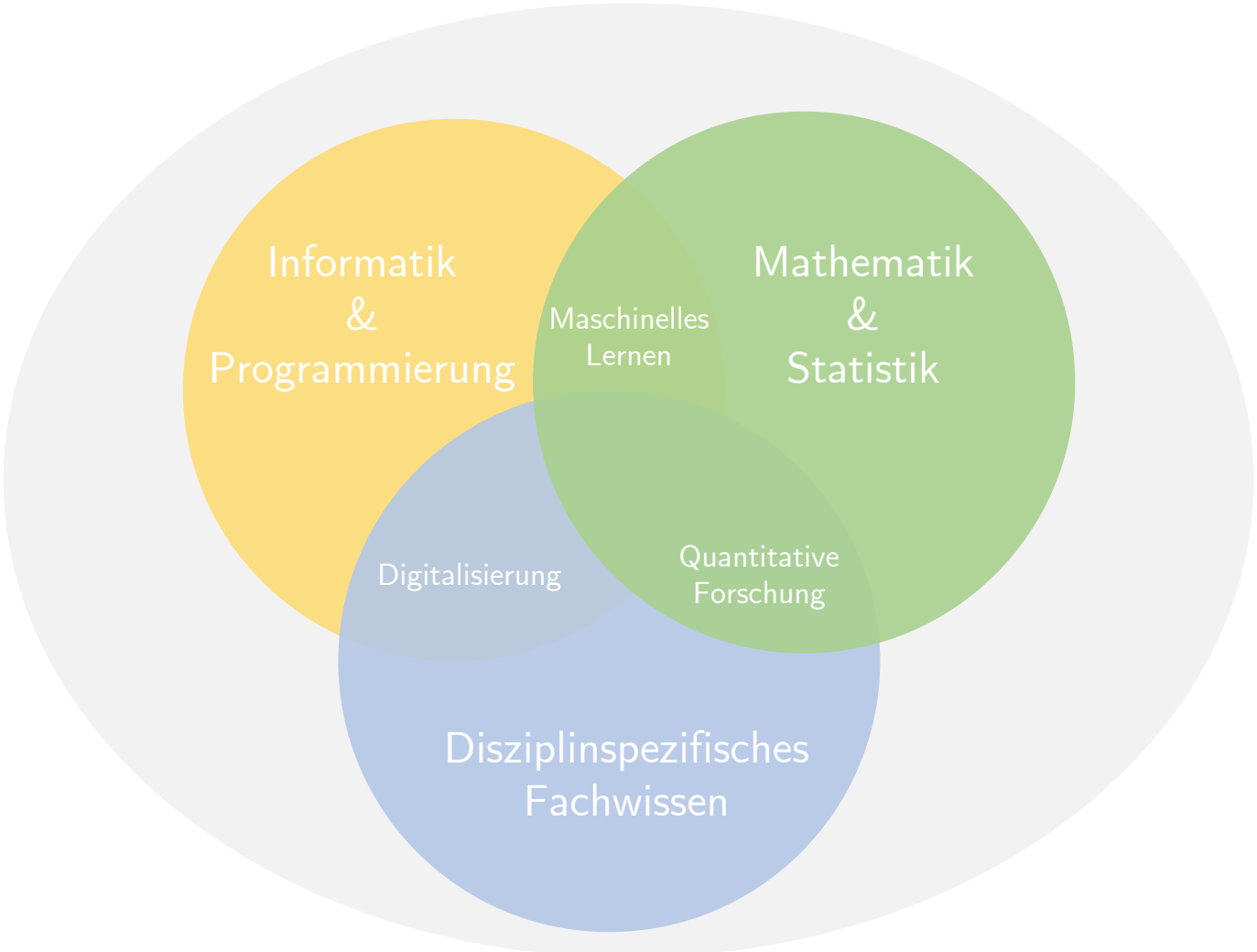
Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Zentrale Komponenten der Datenwissenschaft



Datenanalyse - Überblick

- Wissenschaftliche Daten liegen heutzutage als digitale Daten vor.
- Digitale Daten werden mit Hilfe eines Computers analysiert.
- Zur Analyse von digitalen Daten schreibt man Computerprogramme.
- Diese Computerprogramme heißen Datenanalyseskripte.

Struktur computergestützter Datenanalyse

1. Einlesen und Bereinigen eines digitalen Datensatzes.

2. Berechnung und Visualisierung deskriptiver Statistiken.

3. Probabilistische Modellierung und Inferenz.

4. Dokumentation und Präsentation der Ergebnisse.

Typische Werkzeuge zur Analyse psychologischer Daten

- R (frei, Datenwissenschaft, Statistik, Psychologie)
- Python (frei, Datenwissenschaft, Anwendung)
- Matlab (kommerziell, Engineering, Neuroimaging)

Altmodisch

- SPSS (kommerziell, Sozialwissenschaften, Psychologie)
- JMP (kommerziell, Biologie, Psychologie)
- STATA (kommerziell, Wirtschaftswissenschaften)

Programmiersprachen Trends

PYPL Index (Stand: September 2025)

Worldwide, Sept 2025 :

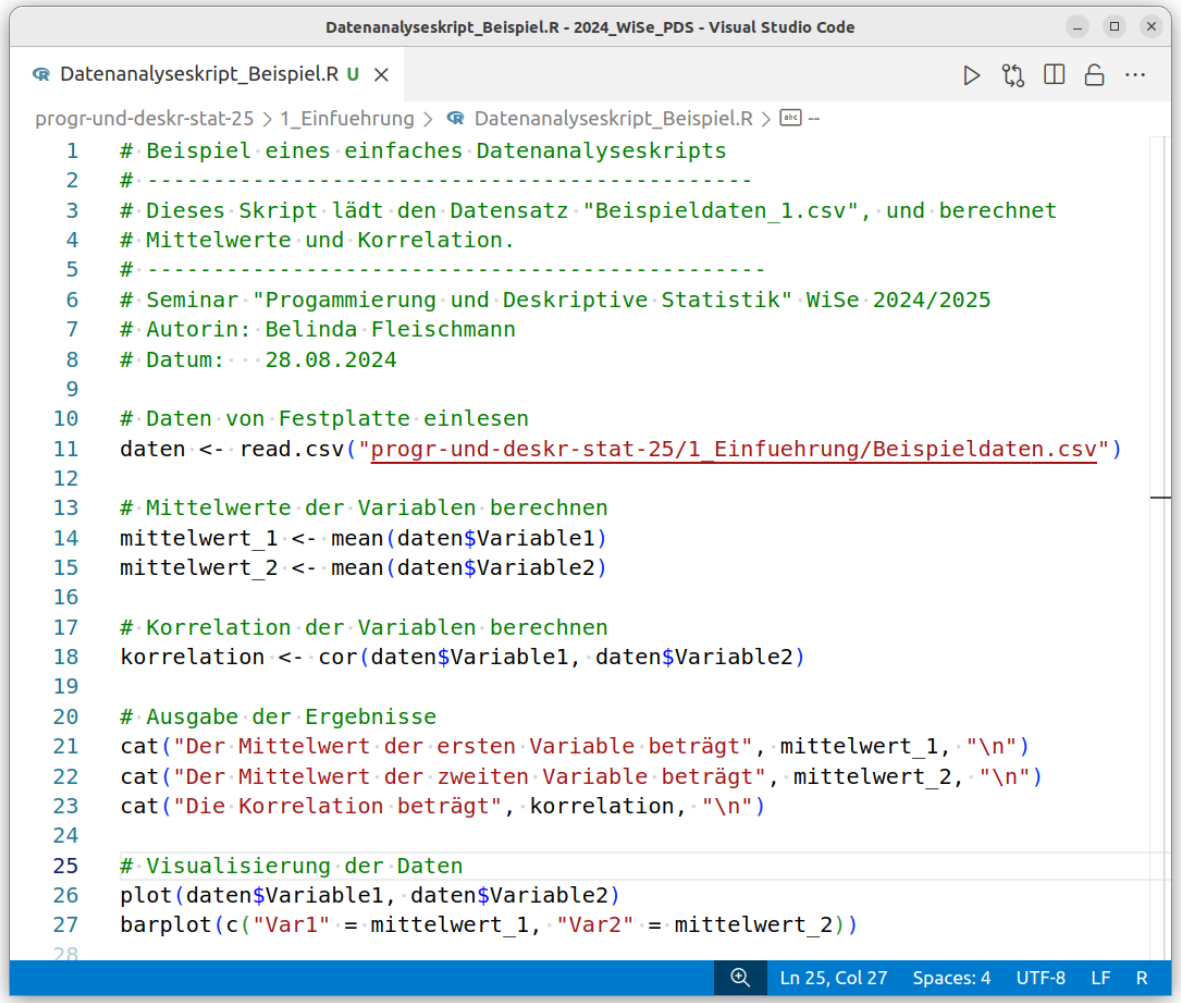
Rank	Change	Language	Share	1-year trend
1		Python	29.69 %	+0.2 %
2		Java	14.72 %	-0.7 %
3	↑	C/C++	9.27 %	+2.5 %
4	↓	JavaScript	6.79 %	-1.4 %
5	↑	R	5.26 %	+0.6 %
6	↓	C#	4.81 %	-1.8 %
7	↑↑↑↑	Objective-C	4.17 %	+1.7 %
8	↓	PHP	3.34 %	-0.8 %
9	↑	Rust	2.73 %	+0.2 %
10	↓	Swift	2.72 %	+0.0 %
11	↓↓↓	TypeScript	2.45 %	-0.5 %
12	↑↑↑	Ada	2.11 %	+1.1 %
13	↓	Go	1.56 %	-0.6 %
14		Matlab	1.49 %	+0.0 %

- PopularitY of Programming Language
- Basierend auf Googlesuchanfragen zu Programmiersprachentutorials

Datenanalyseskripte

- Dokumentation aller Schritte von Rohdaten bis zur Datenvisualisierung.
- Reproduktion wissenschaftlicher Ergebnisse durch Dritte.
- Essentieller Teil wissenschaftlicher Publikationen.
- Essentieller Teil täglicher wissenschaftlicher Arbeit.

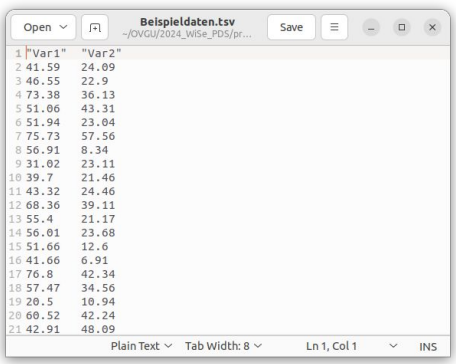
Beispiel

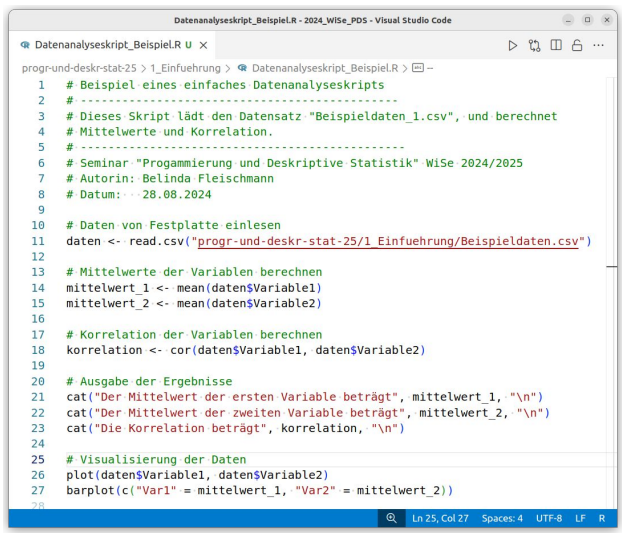

Datenanalyseskript_Beispiel.R - 2024_WiSe_PDS - Visual Studio Code
Datenanalyseskript_Beispiel.R U
progr-und-deskr-stat-25 > 1_Einfuehrung > Datenanalyseskript_Beispiel.R > --
1 # Beispiel eines einfaches Datenanalyseskripts
2 #---
3 # Dieses Skript lädt den Datensatz "Beispieldaten_1.csv", und berechnet
4 # Mittelwerte und Korrelation.
5 #---
6 # Seminar "Progammierung und Deskriptive Statistik" WiSe 2024/2025
7 # Autorin: Belinda Fleischmann
8 # Datum: 28.08.2024
9
10 # Daten von Festplatte einlesen
11 daten <- read.csv("progr-und-deskr-stat-25/1_Einfuehrung/Beispieldaten.csv")
12
13 # Mittelwerte der Variablen berechnen
14 mittelwert_1 <- mean(daten$Variable1)
15 mittelwert_2 <- mean(daten$Variable2)
16
17 # Korrelation der Variablen berechnen
18 korrelation <- cor(daten$Variable1, daten$Variable2)
19
20 # Ausgabe der Ergebnisse
21 cat("Der Mittelwert der ersten Variable beträgt", mittelwert_1, "\n")
22 cat("Der Mittelwert der zweiten Variable beträgt", mittelwert_2, "\n")
23 cat("Die Korrelation beträgt", korrelation, "\n")
24
25 # Visualisierung der Daten
26 plot(daten$Variable1, daten$Variable2)
27 barplot(c("Var1" = mittelwert_1, "Var2" = mittelwert_2))
28
Ln 25, Col 27 Spaces: 4 UTF-8 LF R

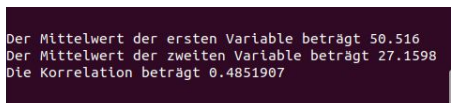
Intuition zur Struktur der Computergestützten Datenanalyse

Datenanalyseskript

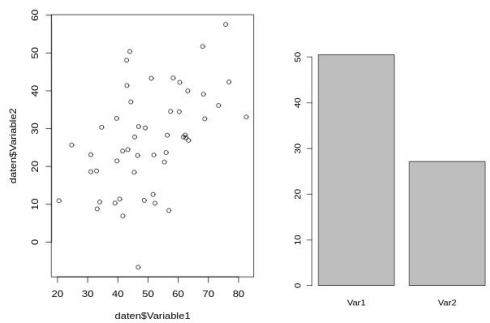
Datensatz







Ergebnisse als Text



Ergebnisse als Visualisierungen

Datenanalyse - Zusammenfassung

- Die Digitalisierung betrifft insbesondere auch die Wissenschaft.
- Forschungsdatenmanagement ist eine akute Herausforderung.
- Programmierung als zentrales Handwerkszeug wissenschaftlicher Arbeit.
- Informatikkenntnisse sind in der Arbeitswelt unverzichtbar.
- Dies gilt auch für Psychotherapeut:innen (z.B. Online-Intervention).

Datenanalyse

Informatik

Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Informatik (engl. Computer Science)

Nach allgemeinem Sprachgebrauch (vgl. Wikipedia) handelt es sich bei der Informatik um die Wissenschaft der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, wobei besonders die automatische Informationsverarbeitung mit Computern betrachtet wird. Die Informatik ist dabei zugleich Grundlagen- und Formalwissenschaft als auch Ingenieurdisziplin

Wikipedia

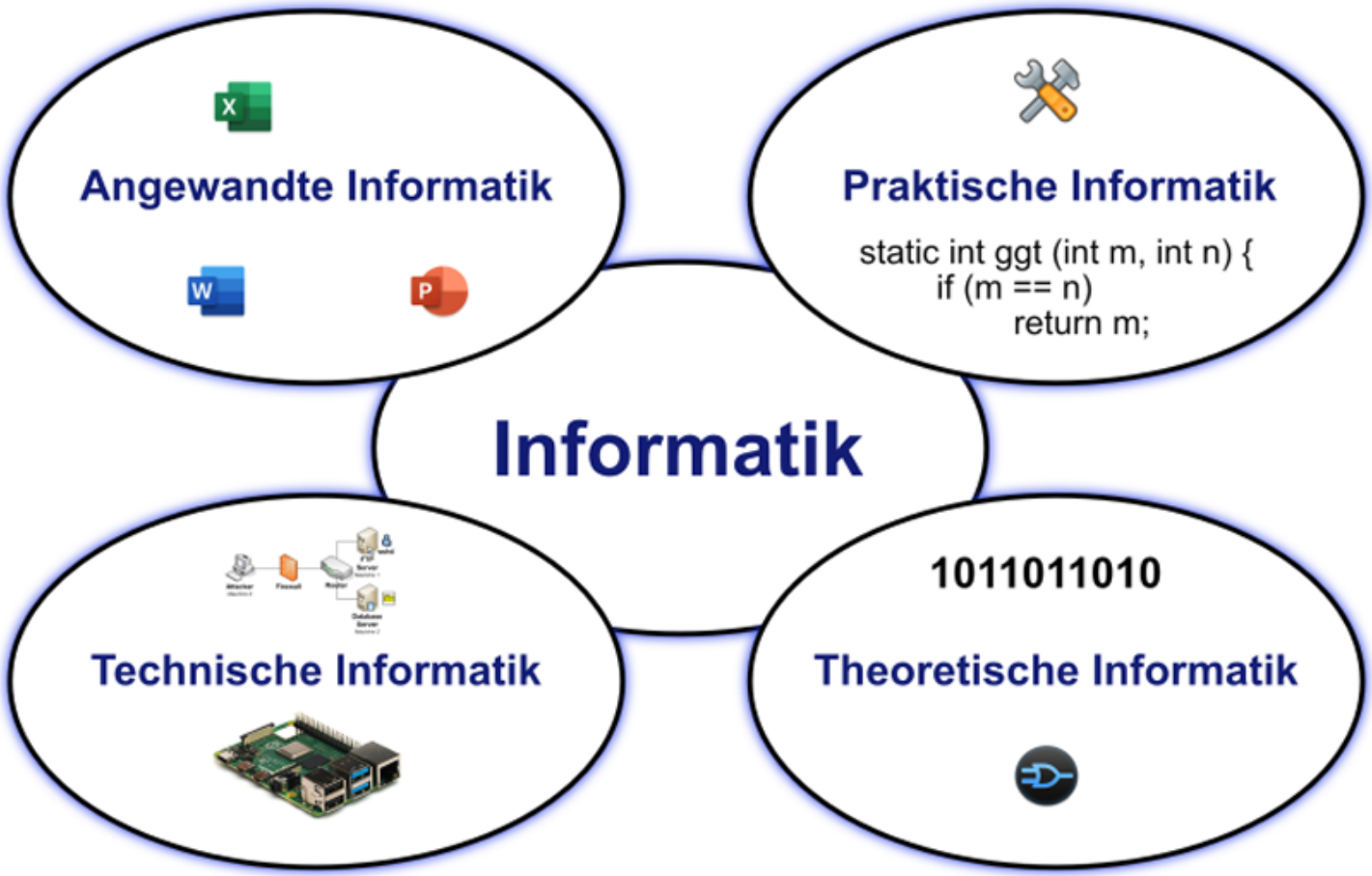
Computer

- Maschinen zum Datenspeichern und Ausführen einfacher Datenoperationen.
- Einfache Operationen mit extrem hoher Geschwindigkeit.
- Universalität durch Speicherung von Daten und Programmen.

Algorithmen und Programme

- *Programme* sind in einer *Programmiersprache* verfasste *Algorithmen*.
- Algorithmen sind Folgen von Anweisungen durchzuführender Operationen.
- Bei Algorithmen unterscheidet man verschiedene Ebenen:
  - Beschreibung (Kochrezept, IKEA Bauanleitung, R Skript)
  - Anweisungen ("Mehl und Wasser vermengen", o - - -, x = c(1,2,3))
  - Durchführung (Kochvorgang, Zusammenbau, R Skript laufen lassen)

Teilgebiete der Informatik



Hattenhauer (2020) Informatik

Teilgebiete der Informatik mit Relevanz für die Psychologie

Technische Informatik

- Mikroprozessortechnik, Rechnerarchitektur, Netzwerktechnik

Theoretische Informatik

- Automatentheorie, Berechenbarkeitstheorie, Komplexitätstheorie

Praktische Informatik

- Programmierung, Algorithmen, Datenbanken

Angewandte Informatik

- Anwendungssoftware, Human-Computer-Interaction, Informatik und Gesellschaft

Spezialgebiete der Informatik mit Relevanz für die Psychologie

Maschinelles Lernen und Künstliche Intelligenz

- Automatisierte Analyse großer Datenmengen und Generierung von Vorhersagen

Computervisualistik

- Bilderkennung und Bildsynthese, Virtual Reality, Augmented Reality
- Psychologische Experimente, Therapien und Trainings

Computerlinguistik

- Spracherkennung und Sprachsynthese
- Chatbots und Assistenzsysteme

Bioinformatik

- Genomik, Bildgebende Verfahren der Medizin
- Methoden für neuropsychologische Forschung und Diagnostik

Datenanalyse

Informatik

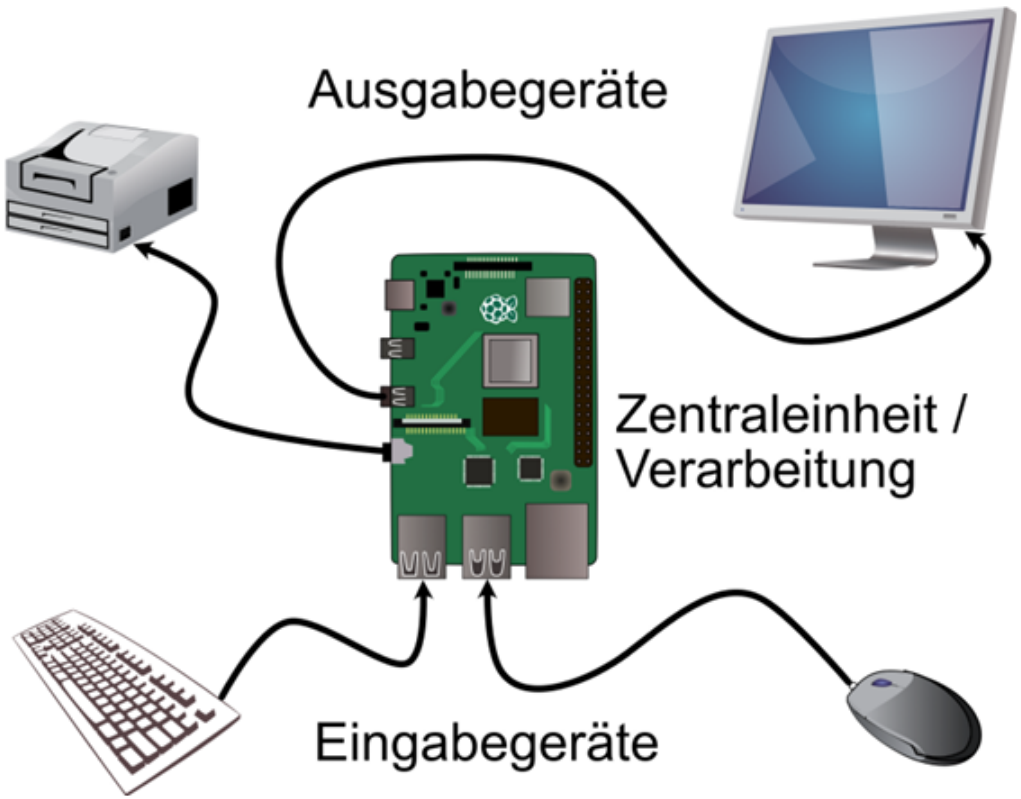
Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

EVA-Prinzip: Eingabe → Verarbeitung → Ausgabe



Hattenhauer (2020) Informatik

Zentraleinheit eines Computers

Die Zentraleinheit, auch Hauptplatine, Motherboard oder Mainboard genannt, ist die zentrale Leiterplatte, auf der alle wichtigen Bauteile wie Prozessor, Speicher und Anschlüsse miteinander verbunden sind.

Hattenhauer (2020) Informatik

Zentraleinheit eines Computers

CPU (Central Processing Unit/Mikroprozessor)

- Rechenwerk, Steuerwerk, und Leitwerk des Systems
- Berechnungen, Steuerung von Datenfluss und Koordination von Abläufen
- Cache (flüchtiger schneller Speicher)
- Intel Core iX, AMD's Ryzen oder Apple Silicon MX Prozessoren

RAM (Random Access Memory)

- Temporärer, flüchtiger Arbeitsspeicher (aktuell benötigte Programme und Daten)
- Begrenzt auf bspw. 16 GB oder 32 GB

Massenspeicher

- Stationärer Speicher des Systems
- SSD (Solid State Drive), Cloudspeicher

GPU (Graphical Processing Unit)

- Leistungsstarke, speziell für Visualisierung optimierte Prozessoren
- Unterstützung der CPU in grafik- oder rechenintensiven Anwendungen, z.b. Training neuronaler Netze

Von Neumann-Architektur

Grundlage der funktionalen Architektur moderner Computer.

Rechner: Steuerwerk, Rechenwerk, Speicher, Eingabewerk, Ausgabewerk.

Zentrale Eigenschaften

- Befehle und Daten werden als Binärzahlen im selben Speicher abgelegt, wodurch Programme während der Ausführung dynamisch verändert oder neu erzeugt werden können.
- Der Speicher ist in gleichgroße nummerierte (adressierte) Zellen unterteilt, deren Inhalt über die Adresse gezielt abgerufen und verändert werden kann.
- Ein Programm ist eine Reihe von aufeinanderfolgenden Befehlen, die in benachbarten Speicherzellen liegen und vom Steuerwerk durch einfaches Erhöhen der Befehlsadresse um eins nacheinander aufgerufen werden.

→ **Diese Architektur eines Rechners impliziert das Grundprinzip der imperativen Programmierung: Befehle werden streng sequentiell abgearbeitet.**

Von Neumann (1945)

Zentrale Eigenschaften (fortgeführt)

- Spezielle Sprungbefehle ermöglichen es, von dieser festen Reihenfolge abzuweichen und an eine andere Stelle im Programm zu springen.
- Grundlegende Befehle umfassen arithmetische Operationen (Addition, Multiplikation), logische Operationen (UND, ODER) sowie Transportbefehle (z.B. Übertragung von Daten vom Eingabewerk in den Speicher oder vom Speicher ins Rechenwerk).
- Sämtliche Informationen (Befehle, Daten, Adressen) werden binär codiert. Enkodierung und Dekodierung erfolgt durch Schaltungstechnik im Rechner.

Von Neumann (1945)

Datenanalyse

Informatik

Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Vom Realweltproblem zum Programm

Realweltproblem

- Das Problem, das mithilfe eines Computers gelöst werden soll.
- z.B. Auswertung von Fragebogendaten einer psychologischen Studie.

Problemspezifikation

- Genaue sprachliche Fassung des Realweltproblems.
- z.B. Methodenteil einer wissenschaftlichen Publikation.

Algorithmus

- Folge von Anweisungen zur Lösung des Problems.
- z.B. Dateneinlesen, deskriptive Statistiken berechnen, T-Test durchführen.

Programm

- Ein Algorithmus, der von einem Computer ausgeführt werden kann.
- Eine in einer Programmiersprache verfasste Textdatei.

Definition (Algorithmus)

Ein *Algorithmus* ist eine Folge von Anweisungen, um aus gewissen Eingabedaten bestimmte Ausgabedaten herzuleiten, wobei folgende Bedingungen erfüllt sein müssen

- *Finitheit.* Die Anweisungsfolge muss in einem endlichen Text vollständig beschrieben sein.
- *Effektivität.* Jede Anweisung muss tatsächlich ausführbar sein.
- *Terminierung.* Der Algorithmus endet nach endlich vielen Anweisungen.
- *Determiniertheit.* Der Ablauf des Algorithmus ist zu jedem Punkt fest vorgeschrieben.

Wenn E die Menge der zulässigen Eingabedaten und A die Menge der zulässigen Ausgabedaten bezeichnet, dann ist ein Algorithmus eine Funktion

$$f : E \to A, e \mapsto f(e) \tag{1}$$

Umgekehrt heißen Funktionen, die durch einen Algorithmus beschrieben werden können, *berechenbare Funktionen*.

Bemerkung

- Effektivität sollte nicht mit Effizienz verwechselt werden.

Eine Programmiersprache

- ... bestimmt die Regeln, denen ein Programm gehorchen muss.
- ... definiert eine Syntax, also Vokabular und Programmaufbau.
- ... definiert Semantik, also die Bedeutung der erlaubten Anweisungen.

```
#if [ -z "$USER_NAME" -o -z "$USER_TYPE" -o -z "$GROUP" ]
if [ -z "$USER_NAME" -o -z "$USER_TYPE" ]
then
#       echo "Please set the user name, type and group"
        echo "Please set the user name and type"
        exit 1
fi

# generate a random password
# -y: include special characters
# -n: include numbers
# -1: one generated passwords per line
#pwgen -y 15 -n 5 -1
echo "Propositions for random passwords to use in next step:"
pwgen -s -n -1 15 5

# add the user
# requires password to be given via input
adduser --firstuid 1000 --lastuid 9999 --no-create-home ${USER_NAME}
```

Programmiersprachen

Maschinensprache

- Elementare Operationsbefehle (z.B. Speichern, Vergleichen, Addieren)
- Elementare Operationsbefehle werden als Binärzahlen kodiert

Addiere Inhalt R1 zu Inhalt R2	$\Rightarrow$ 1001 0010
Erhöhe Inhalt R1 um 1	$\Rightarrow$ 1001 0110
Übertrage Inhalt R1 nach R3	$\Rightarrow$ 0010 0011

- Programme in Maschinensprache heißen *Maschinenprogramme*.
- De facto führt ein Computer nur Maschinenprogramme aus.
- Für Menschen ist die Programmierung in Maschinensprache mühselig.

Höhere Programmiersprache

- An die menschliche Sprache angelehnte Wörter und Sätze
- Interpreter oder Compiler übersetzen Programme in Maschinensprache
- FORTRAN, COBOL, C++, Java, Python, R, u.v.m.

Generationen von Programmiersprachen

1. Generation (1GL)

- Maschinensprachen
- 10110000 01100001 (hexadezimaler Darstellung des Ausdrucks “B0 61”)

2. Generation (2GL)

- Assemblersprachen ab 1950, erste Form der symbolischen Programmierung
- Bspw. “MOV AI, 61H” # Intel-Prozessor-spezifische Sprache

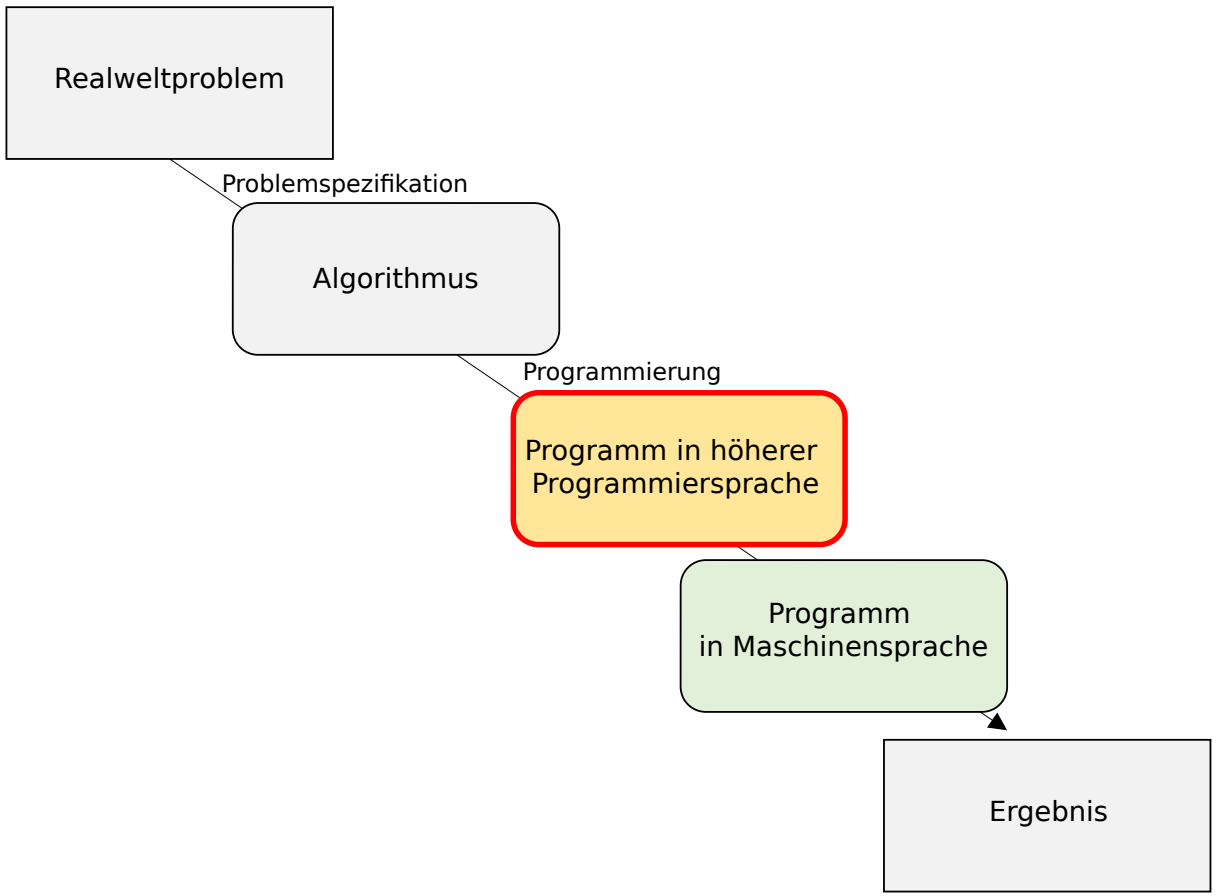
3. Generation (3GL)

- Höhere Programmiersprachen ab 1970 wie FORTRAN, C, C++, Java
- Programmierfreundlich, prozessor-unabhängig

4. Generation (4GL)

- Höhere Programmiersprachen ab 1980 wie Python, Matlab, R
- Codeoverhead Minimisierung, Automation, Flexibilität, Multiparadigmatisch

4GL Programmierung



Imperative Programmierung

- Problemlösungsweg wird als Folge von *Anweisungen (Befehlen)* vorgegeben.
- Befehle verarbeiten Daten, die mithilfe von *Variablen* adressiert werden.
  - **Prozedurale imperative Programmierung**
    - Daten und sie manipulierende Befehle werden separat behandelt.
    - Prozeduren (Funktionen) bilden das zentrale Strukturkonzept.
  - **Objektorientierte imperative Programmierung**
    - Daten und manipulierende Befehle werden als *Objekte* zusammengefasst.
    - Objekte bilden das zentrale Strukturkonzept.
- Praktisch liegen oft Mischformen vor.

Deklarative Programmierung

- Fokus liegt auf der Problembeschreibung, nicht auf dem Lösungsweg.
- Der Programmierende formuliert *was* erreicht werden soll, aber nicht *wie* das Ziel schrittweise erreicht werden soll.
- Das System ermittelt den Lösungsweg automatisch anhand vorgegebener Regeln und Mechanismen.
- Typisch ist die Definition von Beziehungen, Bedingungen oder Zielen statt expliziter Befehlsfolgen und der Computer sucht eigenständig nach einer passenden Lösung.

Compiler und Interpreter

Kompilierte Programmiersprachen

- Gesamter Quellcode wird *vor der Ausführung* in Maschinensprache übersetzt.
- Das Übersetzungsprogramm heißt *Compiler*.
- Der übersetzte Maschinencode wird vom Prozessor ausgeführt.
- Das ausführbare Programm wird nicht übersetzt und läuft schnell.
- Bei Änderungen des Quellcodes muss neu kompiliert werden.
- Beispiele für kompilierte Sprachen sind Java, C, C++.

Interpretierte Programmiersprachen

- Quellcode wird *während der Ausführung* in maschinennahe Sprache übersetzt.
- Das Ausführungsprogramm heißt *Interpreter*.
- Das Programm läuft aufgrund der Interpretation langsamer.
- Bei Änderungen des Quellcodes muss nicht neu interpretiert werden.
- Beispiele für interpretierte Sprachen sind Python und R.

Datenanalyse

Informatik

Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Was ist R?

- Eine Programmiersprache und ein Softwarepaket.
- Entwickelt von Ihaka and Gentleman (1996).
- Freier Dialekt der propietären Software S (Becker, Chambers, and Wilks (1988)).
- Weiterentwickelt und gepflegt durch R Core Team und R Foundation
- Interpretierte imperative 4GL Sprache.
- Optimiert und populär für statistische Datenanalysen.
- Große Community mit etwa 20.000 beigetragenen R Paketen (Erweiterungen)
- Evolviert und konservativ im Kern, konsistent und progressiv in R Paketen.

Wie bekomme ich R?

Über cran.r-project.org die geeignete Version herunterladen und installieren.

CRAN
Mirrors
What's new?
Search
CRAN Team

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2022-06-23, Funny-Looking Kid) R-4.2.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the R project homepage for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

Submitting to CRAN

To "submit" a package to CRAN, check that your submission meets the CRAN Repository Policy and then use the web form.

If this fails, send an email to CRAN-submissions@R-project.org following the policy. Please do not attach submissions to emails, because this will clutter up the mailboxes of half a dozen people.

Was kann R?

- Datensätze laden, manipulieren, und speichern.
- Eine Vielzahl von Berechnungen an verschiedenen Datenstrukturen durchführen.
- Eine Vielzahl statistischer Analysemethoden auf Daten anwenden.
- Datenanalyseskripte schreiben und Abbildungen generieren.
- Präsentationen RMarkdown und Bücher RBookdown erstellen.
- Wissenschaftliche Berichte mit Quarto erstellen.

Was kann R (bisher) nicht so gut?

- In einer ansprechenden Umgebung programmieren ($\Rightarrow$ Visual Studio Code).
- Scientific Computing ($\Rightarrow$ Python, Matlab, Julia).
- Psychologische Experimente programmieren ($\Rightarrow$ Python, Matlab)

Wie bekomme ich Hilfe zu R?

- Während der Programmierung und bei bekanntem Funktionsnamen über die Kommandozeile:

```
?mean                    # Zeigt Hilfe zu der Funktion "mean()"
help(mean)               # Zeigt Hilfe zu der Funktion "mean()"
browseVignettes()        # Zeigt Vignetten aller installierten Pakete im Browser
browseVignettes("knitr") # Zeigt Vignetten des Paktes "knitr"
```

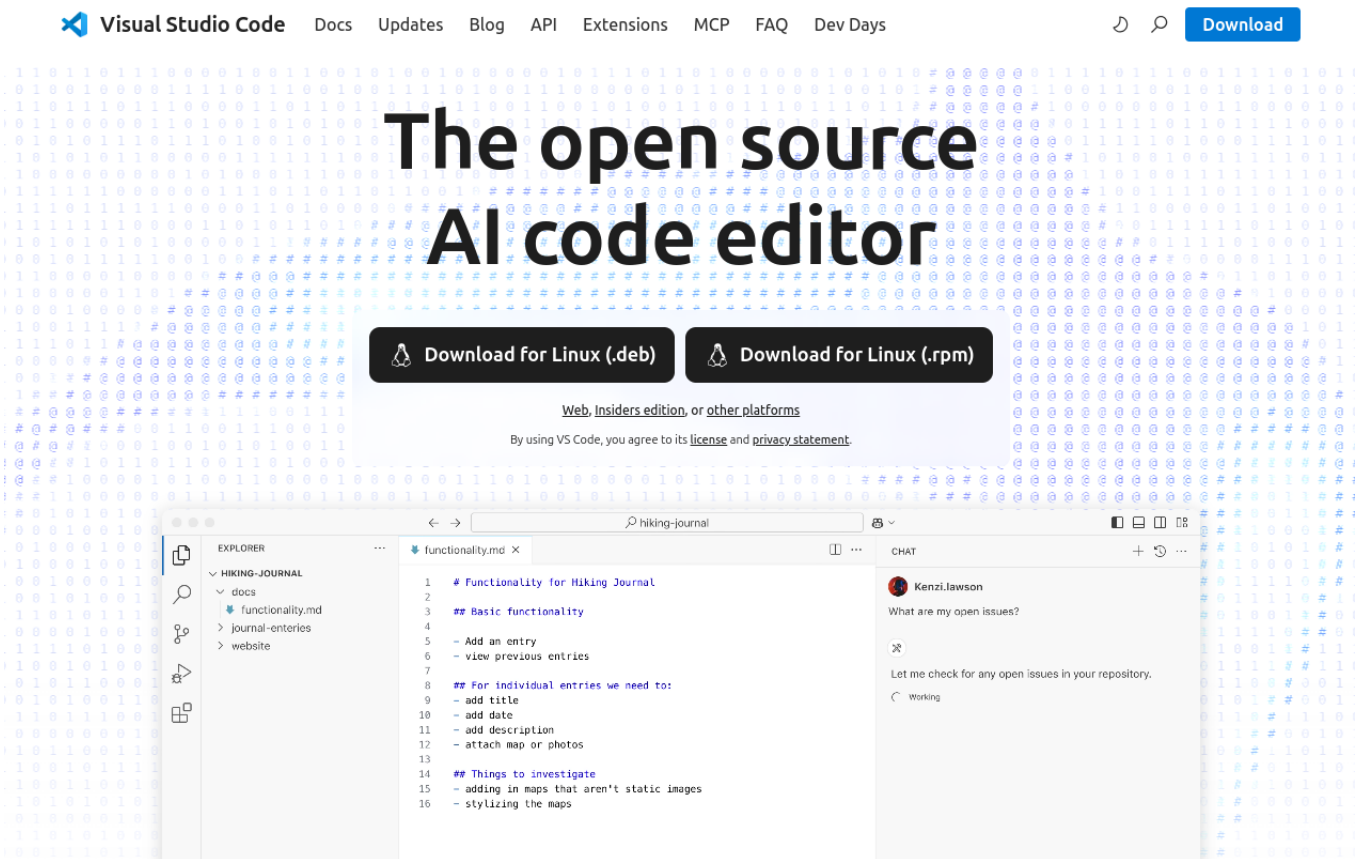
- Googlen
- stackoverflow.com
- r-project.org/help.html
- rseek.org
- rstudio.com/resources/cheatsheets
- r-bloggers.com

Was ist Visual Studio Code (VSCode)?

- VSCode ist eine kostenloser Quelltext-Editor von Microsoft.
- VSCode ist eine Softwareentwicklungsumgebung (Integrated Development Environment, IDE)
- Seit 2015 für Windows, macOS und Linux verfügbar.
- Seit 2018 ist VSCode der beliebteste Editor laut jährlicher stackoverflow Umfragen.
- Ein Microsoftprodukt ist damit auch der beliebteste Editor der Linuxwelt.
- Über Extensions kann VSCode als IDE für beliebige Sprachen genutzt werden.
- Zum Beispiel funktioniert VSCode als IDE für R, Python, Julia, Shell, Quarto, etc.
- VSCode ist Community-based und sehr konfigurierbar.
- VSCode ist über Microsoft's GitHub über Endgeräte synchronisierbar.

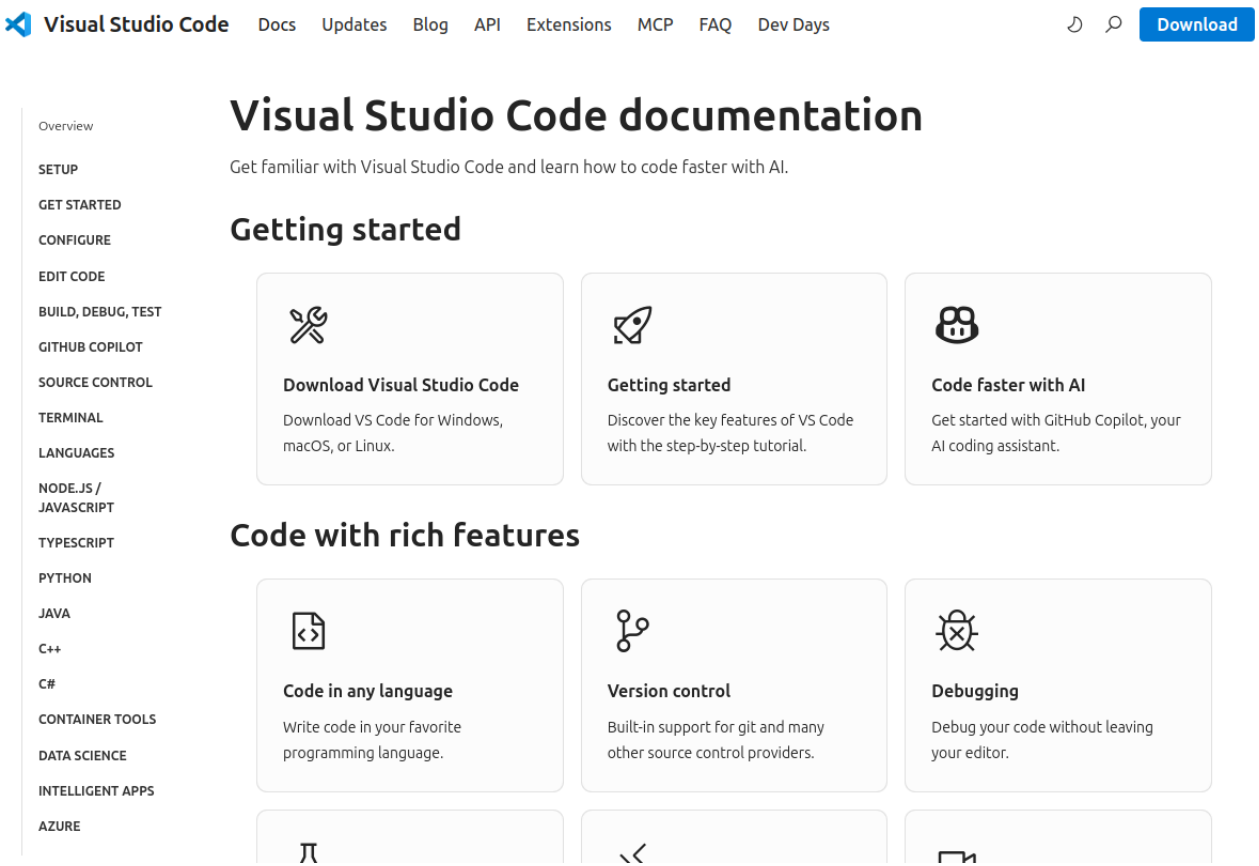
Wie bekomme ich VSCode?

Über https://code.visualstudio.com herunterladen und installieren.



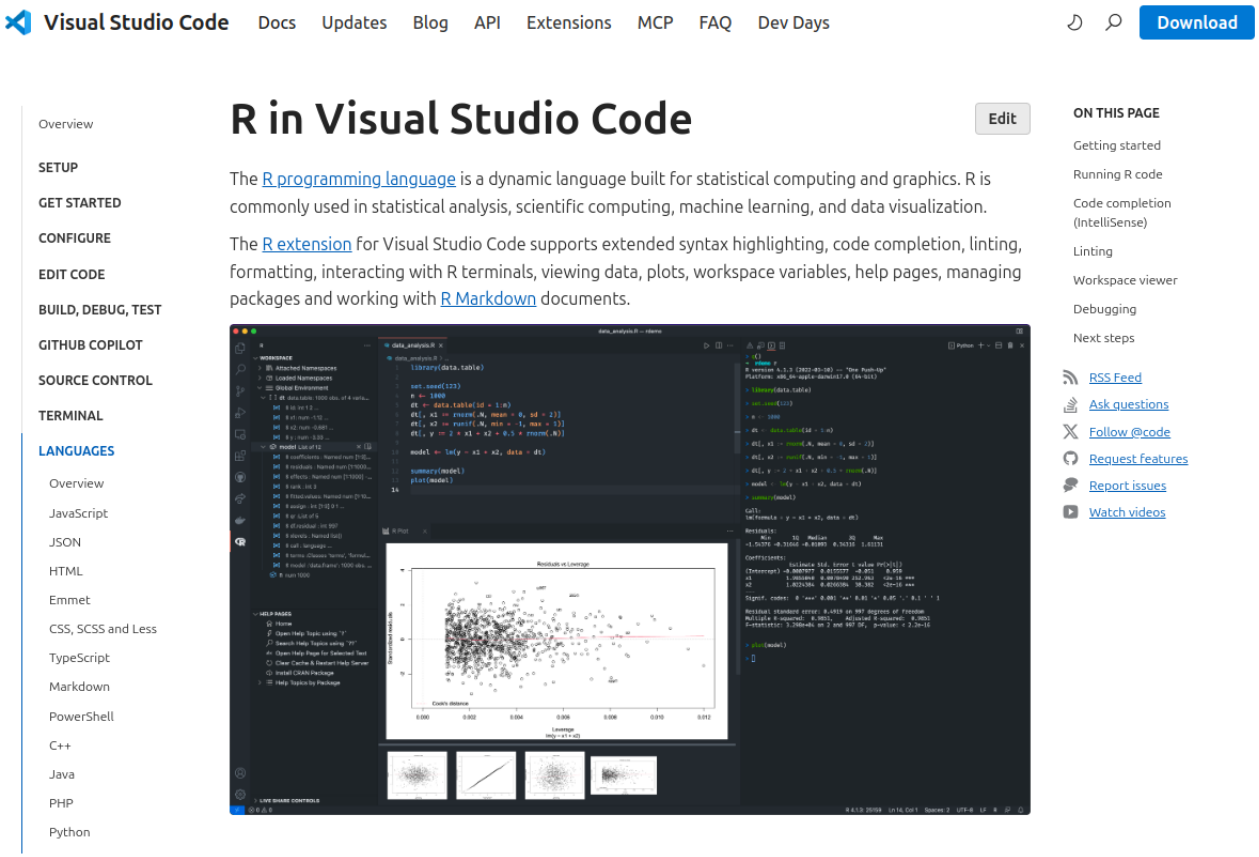
Wie benutze ich VSCode?

Ausführliche Dokumentation mit Anleitungen zur Nutzung: https://code.visualstudio.com/docs



Wie benutze ich R in VSCode?

Anleitungen zum Arbeiten mit R in VSCode: https://code.visualstudio.com/docs/languages/r



Datenanalyse

Informatik

Rechnerarchitektur

Algorithmen und Programme

R und Visual Studio Code

Aufgaben

Aufgaben bis nächste Woche

1. R installieren.
2. VSCode installieren.
3. VSCode für R startklar machen (R in Visual Studio Code).
4. Mit dem eigenen oder Labor-Rechner vertraut machen.
   a) Wie viel Festplatten- und Arbeitsspeicher (RAM) besitzt der Rechner? Notieren Sie die Werte in GB.
   b) Welches Betriebssystem in welcher Version ist installiert? (z.B. Windows 11, macOS 14.5, Ubuntu 22.04)
   c) Erstellen Sie einen Ordner für alle Materialien und Übungen dieses Kurses (z.B. "PDS_2026") an einem selbst gewählten Speicherort auf Ihrer Festplatte (nicht auf dem Desktop oder in Downloads).
   d) Wie lautet der vollständige absolute Pfad zu diesem Kursordner? (z.B. "C:\Users\IhrName\Dokumente\PDS_2026" oder "/home/username/Documents/PDS_2026")
   e) Laden Sie die Dateien "Beispiel_Datenanalyseskript_1.R" und "Beispieldaten.csv" herunter und speichern Sie diese in Ihrem Kursordner.
   f) Testen Sie, mit welchen Programmen sich diese Dateien jeweils öffnen lassen und mit welchen nicht. Welche Programme zeigen den Inhalt sinnvoll lesbar an?

Aufgaben bis nächste Woche

5. Erkunden Sie die VSCode User Interface und bearbeiten Sie nachstehenden Fragen (Tipp: Konsultieren Sie dafür die Online-Dokumentation VSC User interface):
   a) Welche verschiedenen Views (Ansichten) können in der Activity Bar (vertikale Leiste ganz links) ausgewählt werden? Listen Sie mindestens fünf auf.
   b) Wählen Sie in der *Activity Bar* den *Explorer* aus. Wie können Sie Ihren Kursordner in VSCode öffnen, sodass dessen Inhalt in der Primary Sidebar (Hauptseitenleiste) dauerhaft sichtbar ist?
   c) Installieren Sie die R Extension für VSCode, falls noch nicht geschehen. Wählen Sie in der *Activity Bar* die *R Extension* aus. Welche Bereiche und Funktionen werden nun in der Primary Sidebar angezeigt?
   d) Standardmäßig befindet sich das Panel (mit Terminal, R-Console, etc.) am unteren Bildschirmrand. Wie können Sie es so konfigurieren, dass das Panel stattdessen rechts (oder links) vertikal angeordnet ist?
   e) Wie können Sie eine Secondary Sidebar öffnen und wofür könnte diese nützlich sein?

Referenzen

Becker, Richard A., John M. Chambers, and Allen Reeve Wilks. 1988. *The New S Language: A Programming Environment for Data Analysis and Graphics*. Reprint. London: Chapman & Hall.

Ihaka, Ross, and Robert Gentleman. 1996. "R: A Language for Data Analysis and Graphics." *Journal of Computational and Graphical Statistics* 5 (3): 2999–2314.

Von Neumann, John;United States Army Ordnance Department ;University of Pennsylvania. 1945. "First Draft of a Report on the EDVAC." Digital Book. Moore School of Electrical Engineering, University of Pennsylvania. 1945. https://library.si.edu/digital-library/book/firstdraftofrepo00vonn.