

# Praktische Informatik 1

## Organisatorisches und Einführung

Thomas Röfer

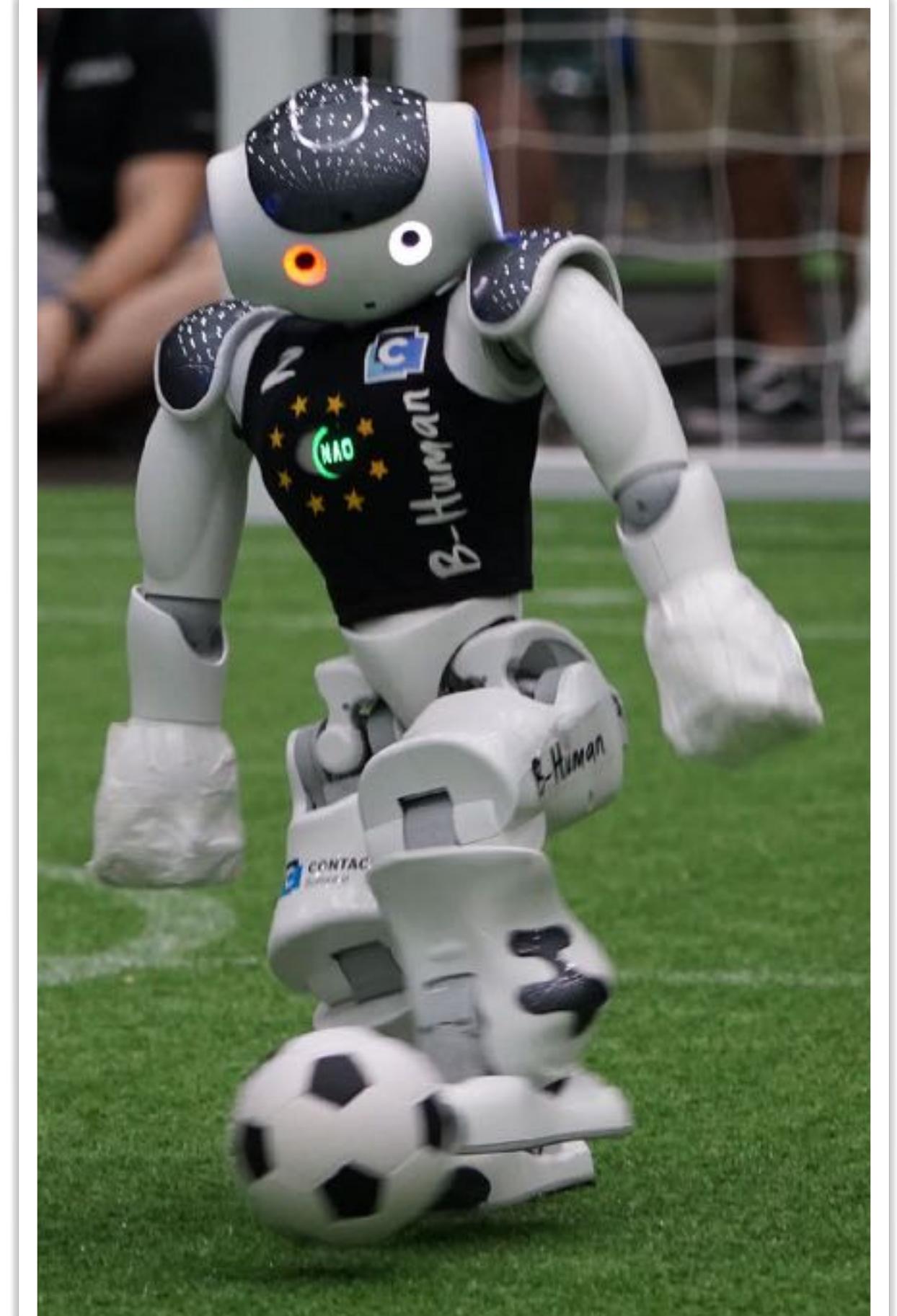
Cyber-Physical Systems  
Deutsches Forschungszentrum für  
Künstliche Intelligenz

Multisensorische Interaktive Systeme  
Fachbereich 3, Universität Bremen

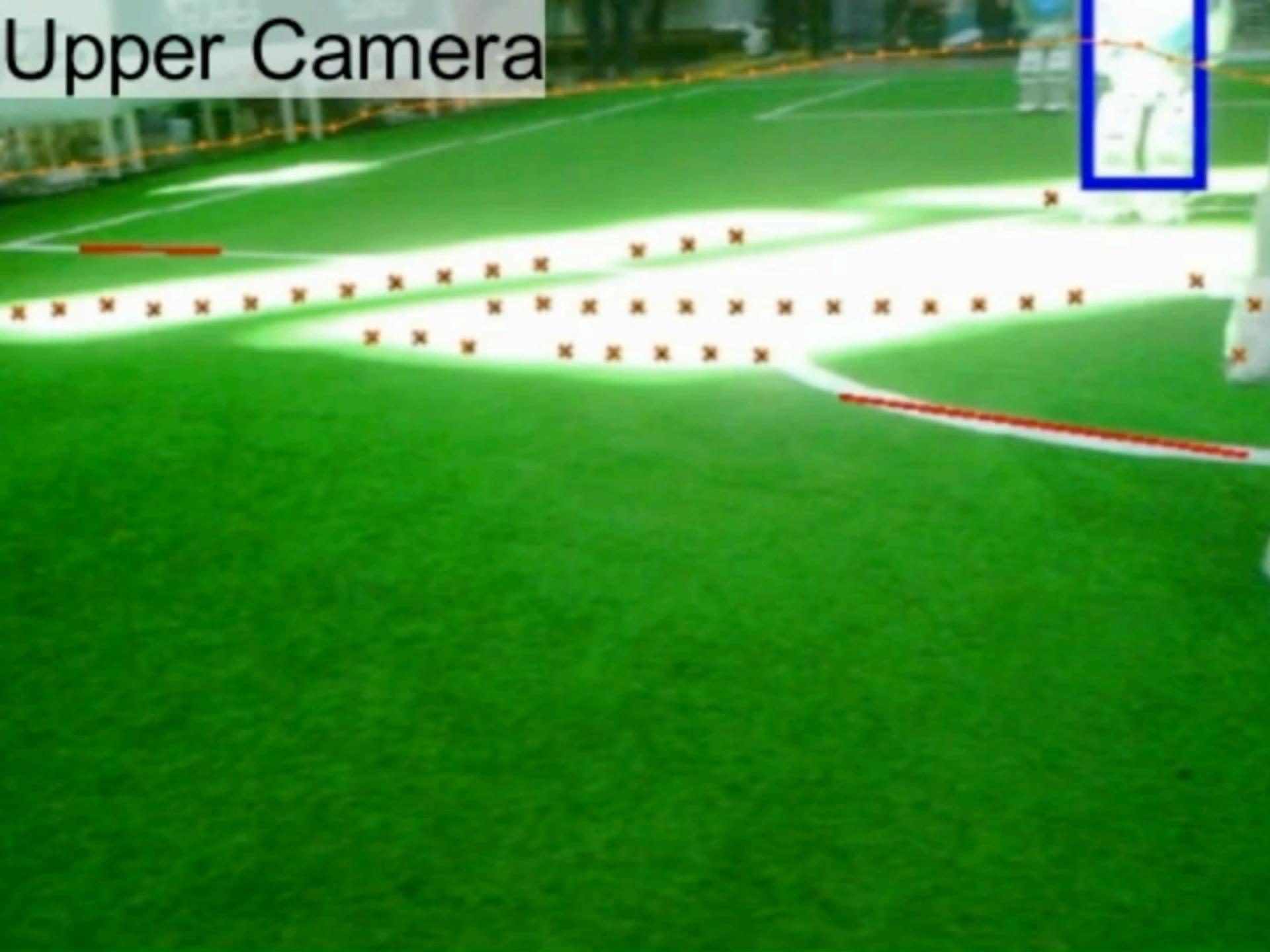


## Über mich...

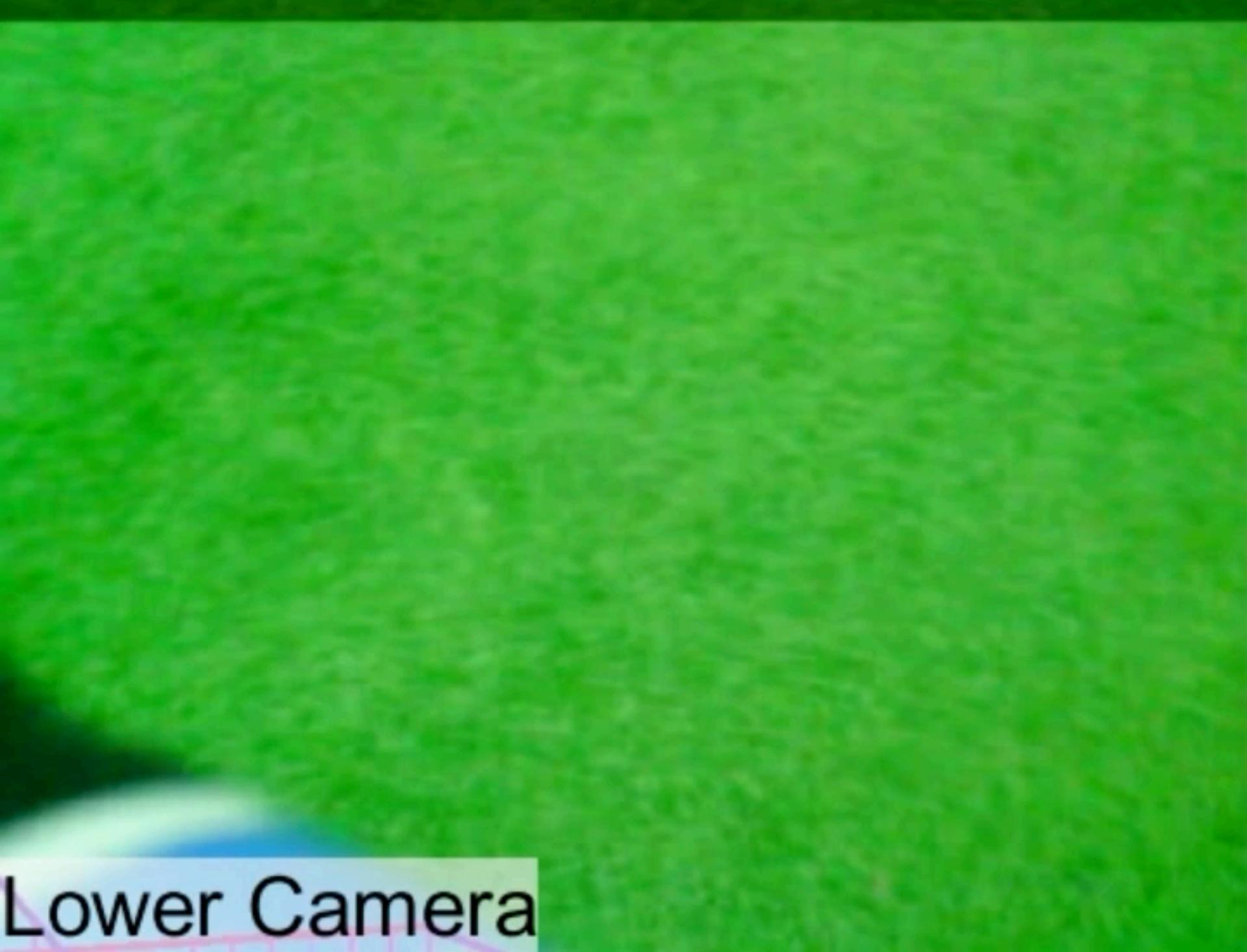
- Senior Researcher im Forschungsbereich „Cyber-Physical Systems“ (Leiter: Rolf Drechsler) im Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI)
- Mitglied der Arbeitsgruppe „Multisensorielle interaktive Systeme“ (Leiter: Udo Frese) im Fachbereich 3 der Universität Bremen
- Leite mit Tim Laue studentisches Projekt „B-Human“
  - 9x Weltmeister und 12x Meister in europäischen Wettbewerben in der RoboCup Standard Platform League
  - ~140000 Zeilen C++



Upper Camera



World Model



Lower Camera



Live Camera

# Ziele: Programmierung

- Verstehen der Konzepte
  - Zustandsbasierte Programmierung
  - objektorientierte Programmierung
- Erlernen der Fertigkeiten
  - Programmieren
  - Debuggen
- ... am Beispiel der Programmiersprache

```
abstract class PI1Game extends Game
{
    static void main()
    {
        var obj = new GameObject(2, 1, 0, "laila");
        while (true) {
            obj.setRotation(obj.getRotation() + 1);
            sleep(500);
        }
    }
}
```



# Programmierung in Java



- BlueJ
- Lehr-Entwicklungsumgebung
  - Eingebauter Editor
  - Code direkt ausführbar
  - Debugger
- Läuft überall (in Java geschrieben)
- Basis des Buchs zur Vorlesung

The screenshot shows the BlueJ IDE interface. At the top, there's a toolbar with buttons for 'Neue Klasse...', 'Übersetzen', 'Teamwork' (with 'Share...'), and 'Testing' (with 'Tests starten'). Below the toolbar is a UML class diagram for a game application. It features an abstract class 'Game' with a generalization arrow pointing to an abstract class 'PI1Game'. A concrete class 'GameObject' is shown below 'Game'. To the right of the diagram is a code editor window titled 'PI1Game - PI1Game'. The code is as follows:

```

1 // Importieren der VK_*-Tastenkonstanten
2 import static java.awt.event.KeyEvent.*;
3
4 /**
5  * Dies ist die Hauptklasse eines Spiels. Sie enthält die Hauptmethode, die zum
6  * Starten des Spiels aufgerufen werden muss.
7  *
8  * @author Thomas Röfer
9  */
10 abstract class PI1Game extends Game
11 {
12     /** Das Spiel beginnt durch Aufruf dieser Methode. */
13     static void main()
14     {
15
16     }
17 }

```

The code editor has a status bar at the bottom that says 'Klasse übersetzt - keine Syntaxfehler' and 'gespeichert'. In the top right corner of the slide, there is a small image of a blue jay in flight.

# Ziele: Dokumentenerstellung mit LaTeX

- Quelltext → Compiler → Dokument
- Strukturierte Dokumente
- Vorteile
  - Keine Gefahr „defekter“ Dokumente
  - Viele Texte in der Mathematik und Informatik werden mit  $\text{\LaTeX}$  erstellt
- Lösungsvorschläge müssen mit  $\text{\LaTeX}$  erstellt und in Form von Quelltexten zusammen mit den Java-Quelltexten abgegeben werden

```
\documentclass{pi1}
\begin{document}

\maketitle{1}{O. Schlau}{R. Ratlos}

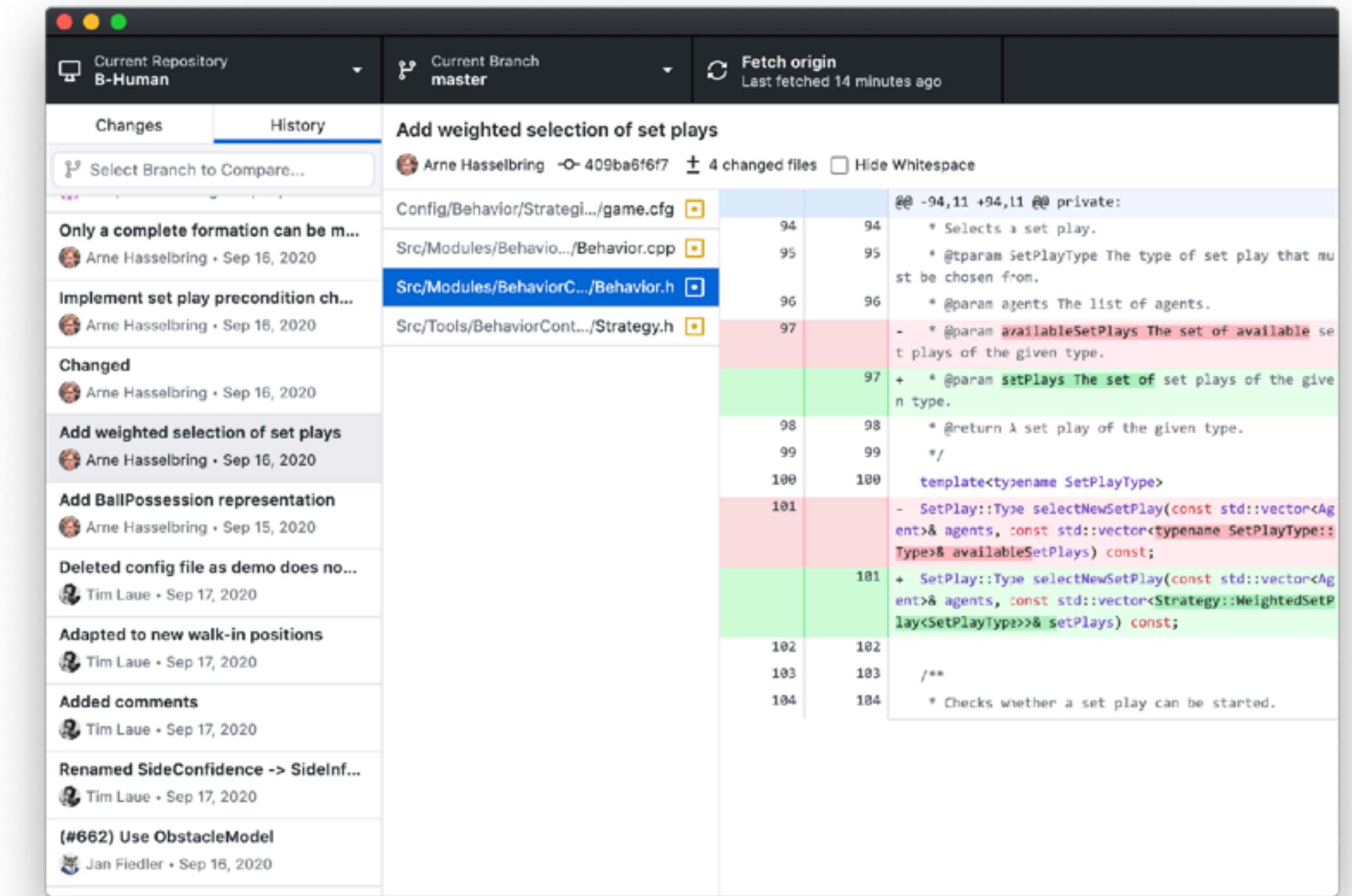
\section{Die Summe über $X$}

\begin{equation}
f(X) = \sum_{i=1}^n X_i
\end{equation}

\end{document}
```

# Ziele: Versionsverwaltung mit git

- Änderungshistorie verwalten
  - Wer hat was wann wie verändert?
  - Vergisst nie etwas, d.h. alle alten Versionen der Dateien sind noch da!



```

Changes History
Select Branch to Compare...
Only a complete formation can be m...
Implement set play precondition ch...
Changed
Add weighted selection of set plays
Add BallPossession representation
Deleted config file as demo does no...
Adapted to new walk-in positions
Added comments
Renamed SideConfidence -> SideIn...
(#662) Use ObstacleModel
Arne Hasselbring -O 409ba6f6f7 ± 4 changed files Hide Whitespace
Config/Behavior/Strategi.../game.cfg
Src/Modules/Behavio.../Behavior.cpp
Src/Modules/BehaviorC.../Behavior.h
Src/Tools/BehaviorCont.../Strategy.h
94 94 @@ -94,11 +94,11 @@ private:
    * Selects a set play.
95 95 * @param SetPlayType The type of set play that mu-
    st be chosen from.
96 96 * @param agents The list of agents.
97 97 - * @param availableSetPlays The set of available se-
    t plays of the given type.
98 98 + * @param setPlays The set of set plays of the give-
    n type.
99 99 * @return A set play of the given type.
100 100 */
101 101 template<typename SetPlayType>
102 102 - SetPlay::Type selectNewSetPlay(const std::vector<Ag-
    ents>& agents, const std::vector<typename SetPlayType::
    Type>& availableSetPlays) const;
103 103 + SetPlay::Type selectNewSetPlay(const std::vector<Ag-
    ents>& agents, const std::vector<Strategy::WeightedSetP-
    lays<SetPlayType>>& setPlays) const;
104 104 /**
    * Checks whether a set play can be started.

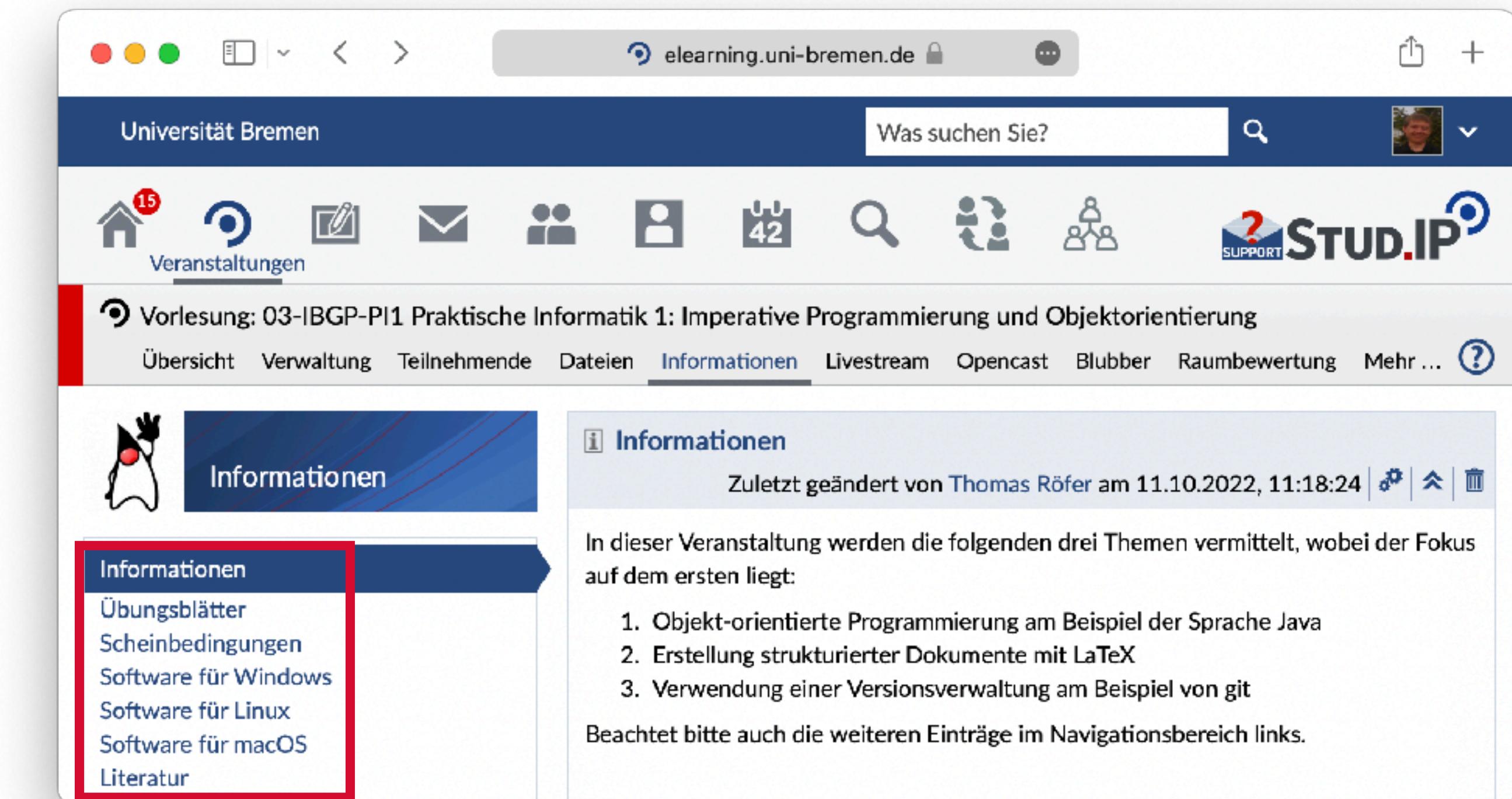
```

- Strukturierte Zusammenarbeit über das  GitLab der Informatik
- DropBox usw. eignen sich dagegen für Programmierprojekte nicht
  - Fehler würden sofort weiterverteilt, so dass andere nicht arbeiten können

# Informationen zu PI-1 im Stud.IP

- **elearning.uni-bremen.de**: 03-IBGP-PI1

- Teilnehmende|Gruppen: Praktikumslisten
- Dateien|Praktikum XX...: Ordner mit Praktikumsmaterialien und Abgabeordnern
- Informationen: Material zur Vorlesung
- E-Mail: An Tutor:innen oder (als letzte Möglichkeit) mich (unter Teilnehmende)



The screenshot shows a web browser window for [elearning.uni-bremen.de](https://elearning.uni-bremen.de). The page title is "Universität Bremen". The navigation bar includes links for Veranstaltungen (with 15 notifications), Übersicht, Verwaltung, Teilnehmende, Dateien, **Informationen**, Livestream, OpenCast, Blubber, Raumbewertung, and Mehr... A user profile picture is visible in the top right.

The main content area displays information for a lecture titled "Vorlesung: 03-IBGP-PI1 Praktische Informatik 1: Imperative Programmierung und Objektorientierung". The "Informationen" tab is selected. On the left, there's a sidebar with a "Informationen" icon and a list of links: Übungsblätter, Scheinbedingungen, Software für Windows, Software für Linux, Software für macOS, and Literatur. The "Informationen" section on the right contains a message from Thomas Röfer last updated on 11.10.2022 at 11:18:24. It states: "In dieser Veranstaltung werden die folgenden drei Themen vermittelt, wobei der Fokus auf dem ersten liegt:" followed by a numbered list: 1. Objekt-orientierte Programmierung am Beispiel der Sprache Java, 2. Erstellung strukturierter Dokumente mit LaTeX, 3. Verwendung einer Versionsverwaltung am Beispiel von git. A note at the bottom says: "Beachtet bitte auch die weiteren Einträge im Navigationsbereich links."

- Wird üblicherweise per E-Mail (nicht per Stud.IP-Nachricht) beantwortet!
- Alle Dinge im Zusammenhang mit FlexNow/Pabo kann nur das Prüfungsamt lösen!

## Zusatzübung Programmieren (03-IBFW-ZÜP)

- Zusatzübung für Erstsemester **ohne** Programmierkenntnisse
- Von Karsten Hölscher
- Eigenständiger Kurs, unabhängig von PI-1
  - Keine Fragen zum Vorlesungsstoff oder den Übungen von PI-1
- Mehr Aufgaben, mehr Übung, mehr Feedback
- Bei Anmeldung Fragebogen zu Vorkenntnissen



## Coding-Support-Projekt ( 03-IB-CSP )

- Studentisches Angebot für Studierende mit **wenig** Programmierkenntnissen
- Von Johanna Götz, Jethro Bartel und Pascal Himmelmann
- Wir wollen mit euch gemeinsam in einem lockeren Rahmen gemeinsam programmieren, um eure Kenntnisse zu vertiefen
- Genaue Schwerpunkte und Inhalte werden mit den Teilnehmenden abgesprochen
- Fragen gerne an : **[coding-projekt@groups.uni-bremen.de](mailto:coding-projekt@groups.uni-bremen.de)**

# Grundlagen: Analytical Engine (Charles Babbage, 1834)

- Aufbau
  - Prozessoreinheit
  - Speichereinheit
  - Programmsteuerung
- Programme
  - sind Sequenzen von Befehlen,
  - werden nacheinander vom Prozessor verarbeitet,
  - werden auf Daten im Speicher angewendet

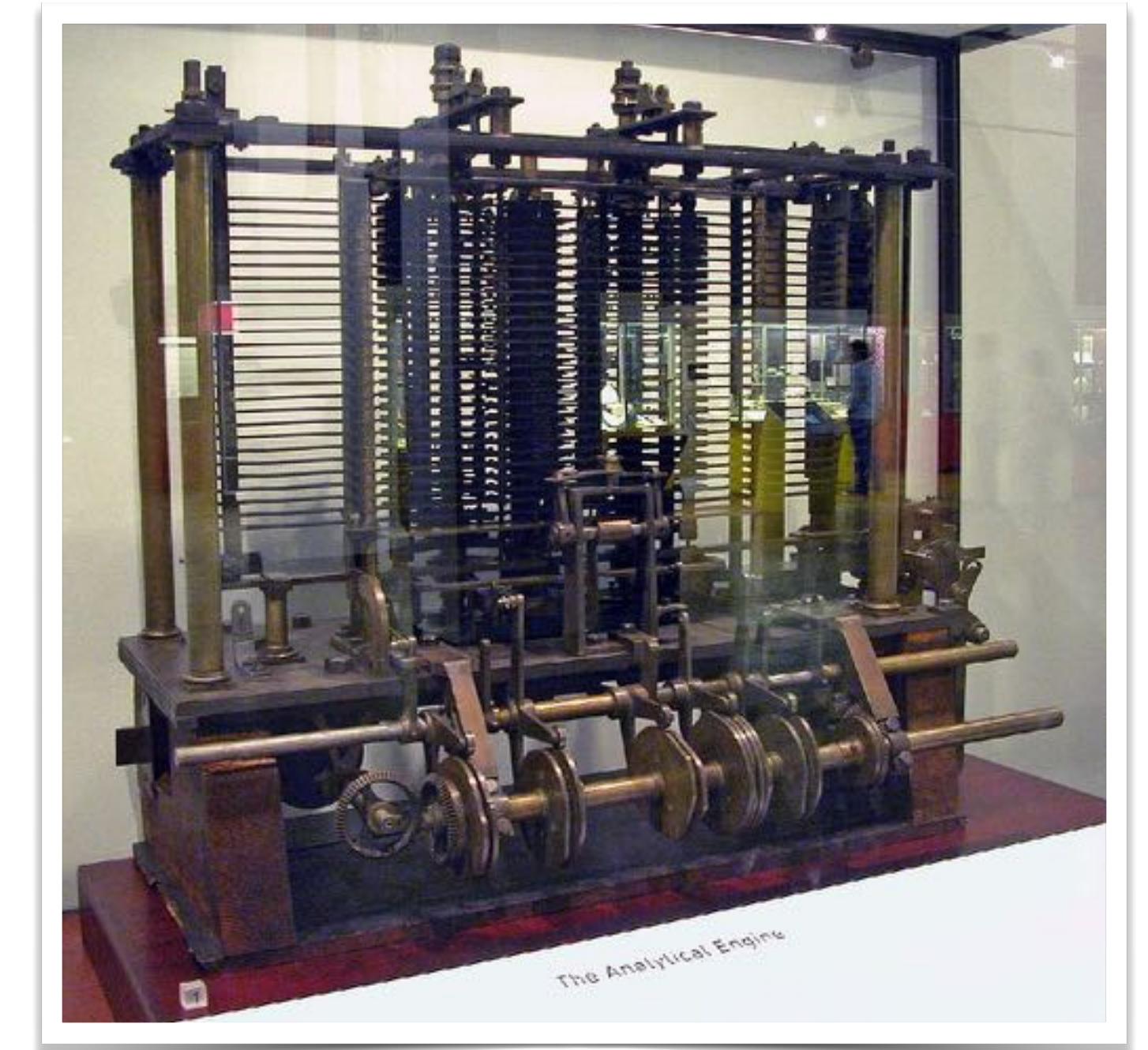
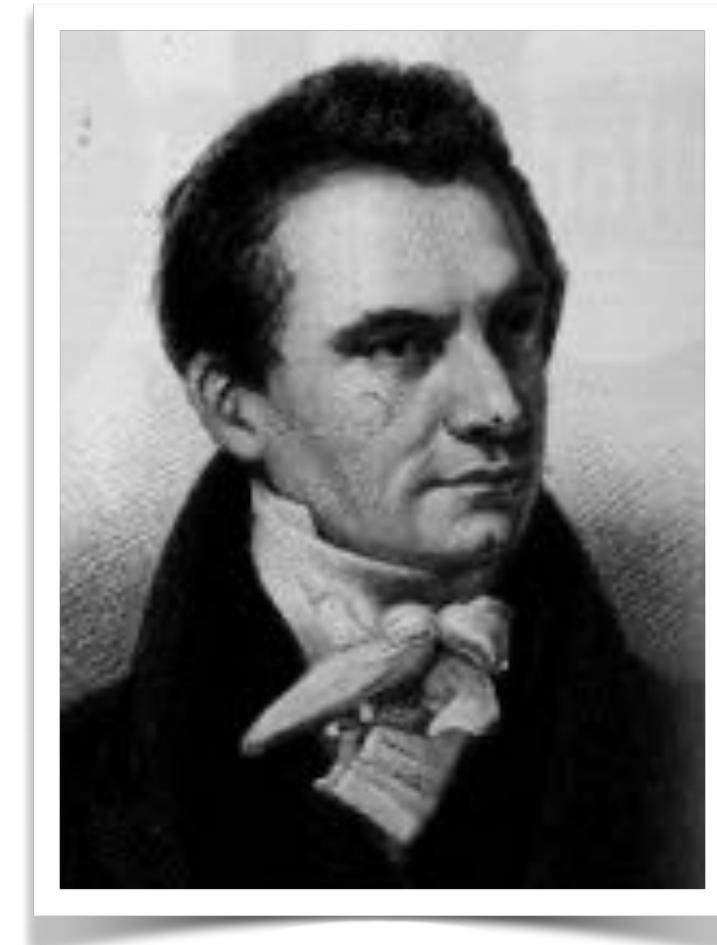


Bild: ©Bruno Barral (ByB)

# Grundlagen

- Erste Rechner
  - Zuse Z3 (1941)
  - Mark 1 (1944)
  - ENIAC (1946)
  - turingmächtig
- von-Neumann-Architektur (1945)
  - “First Draft of a Report on the EDVAC”



Bild: ©Wolfgang Hunscher

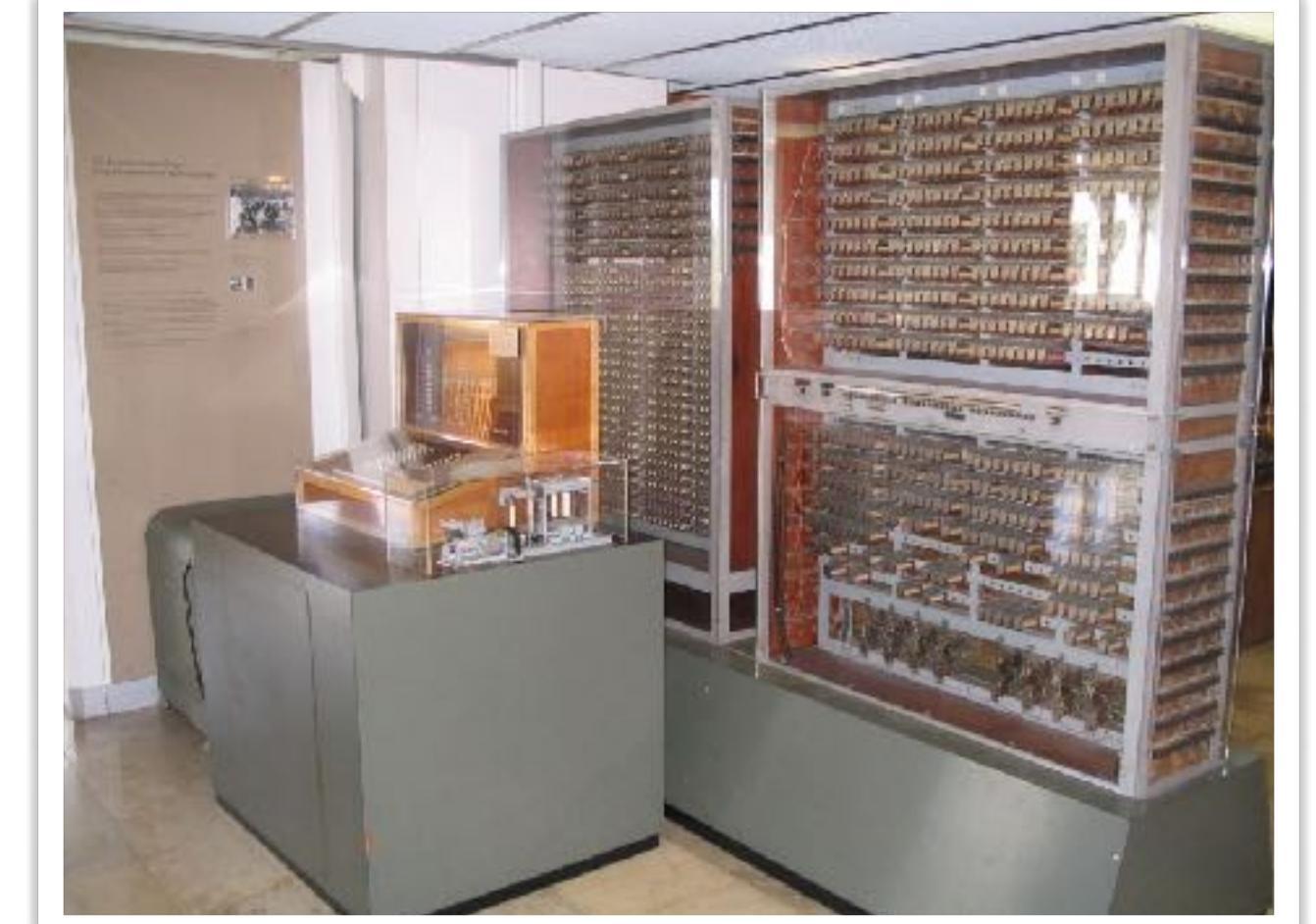
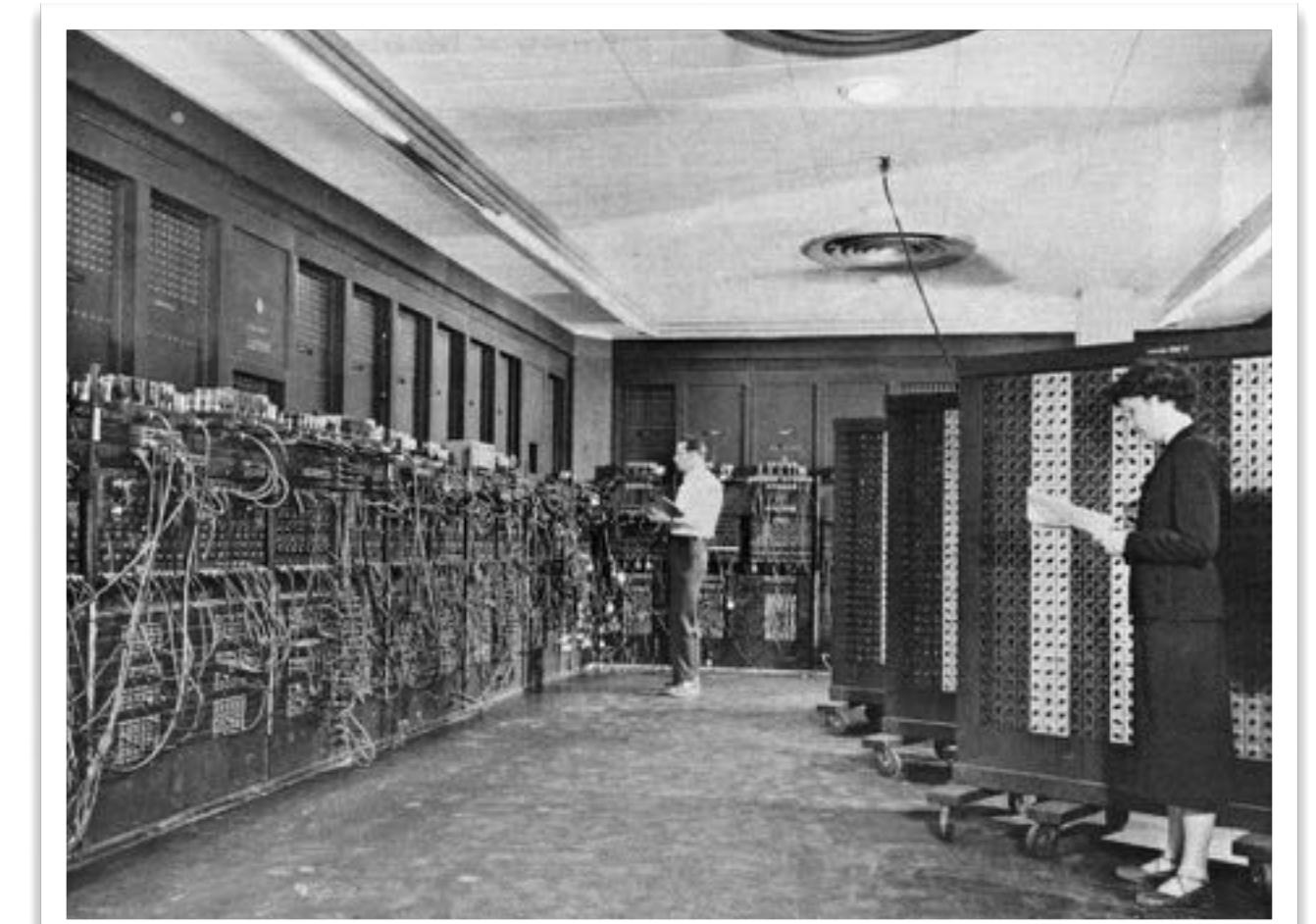
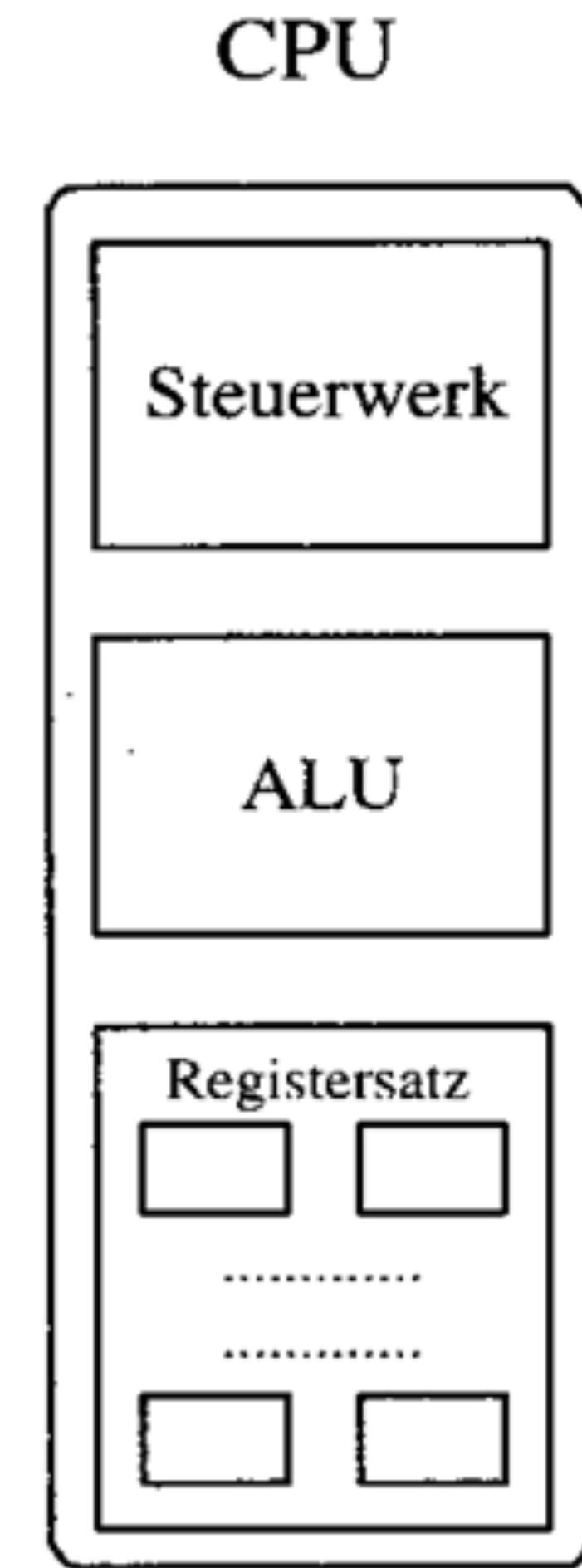


Bild: ©Venusianer



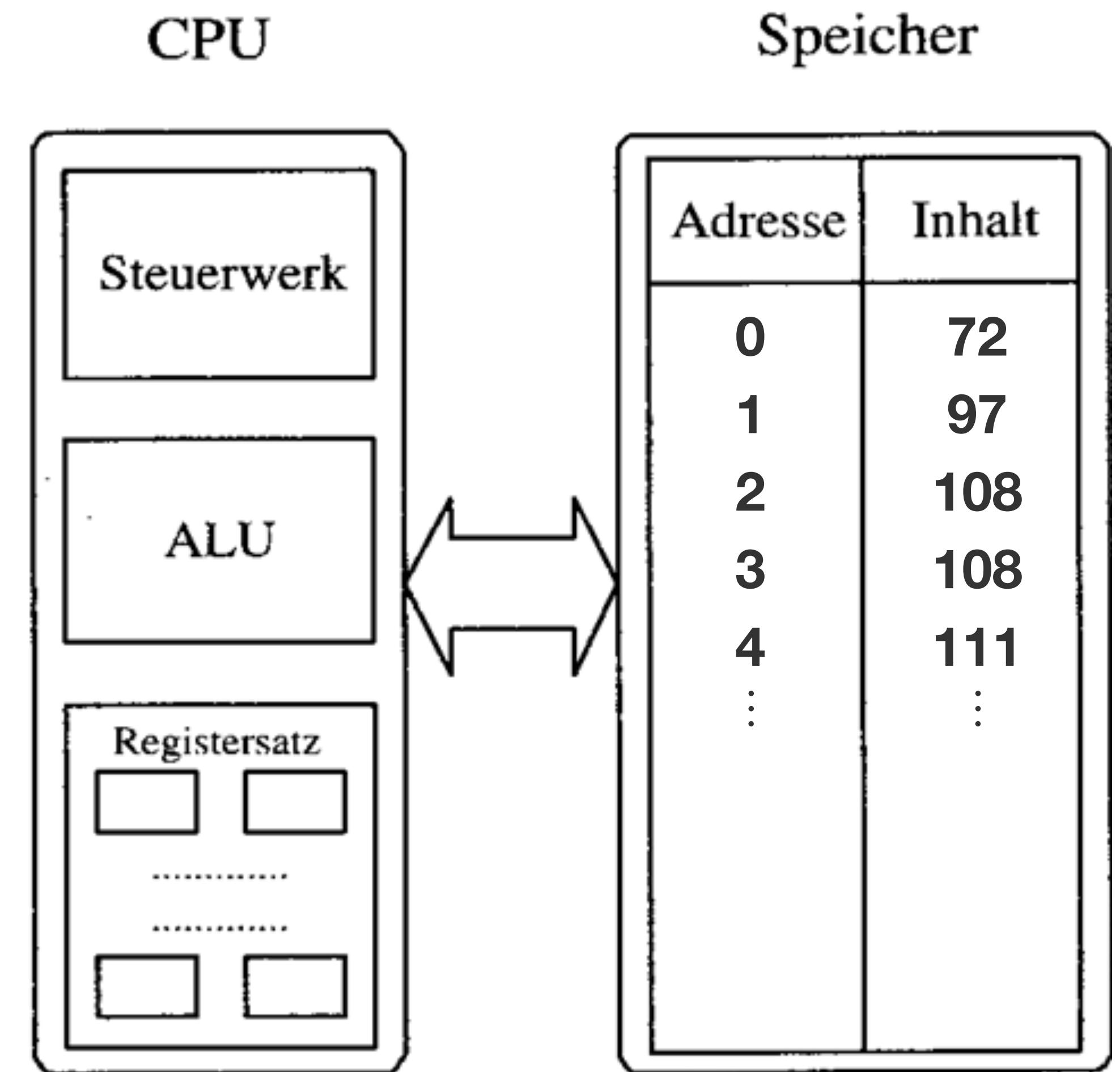
## von-Neumann-Architektur: Prozessor (CPU)

- Steuerwerk
  - Führt Programme aus
- Arithmetisch-logische Einheit (ALU)
  - Rechnet
- Registersatz
  - Speichert Daten im Prozessor



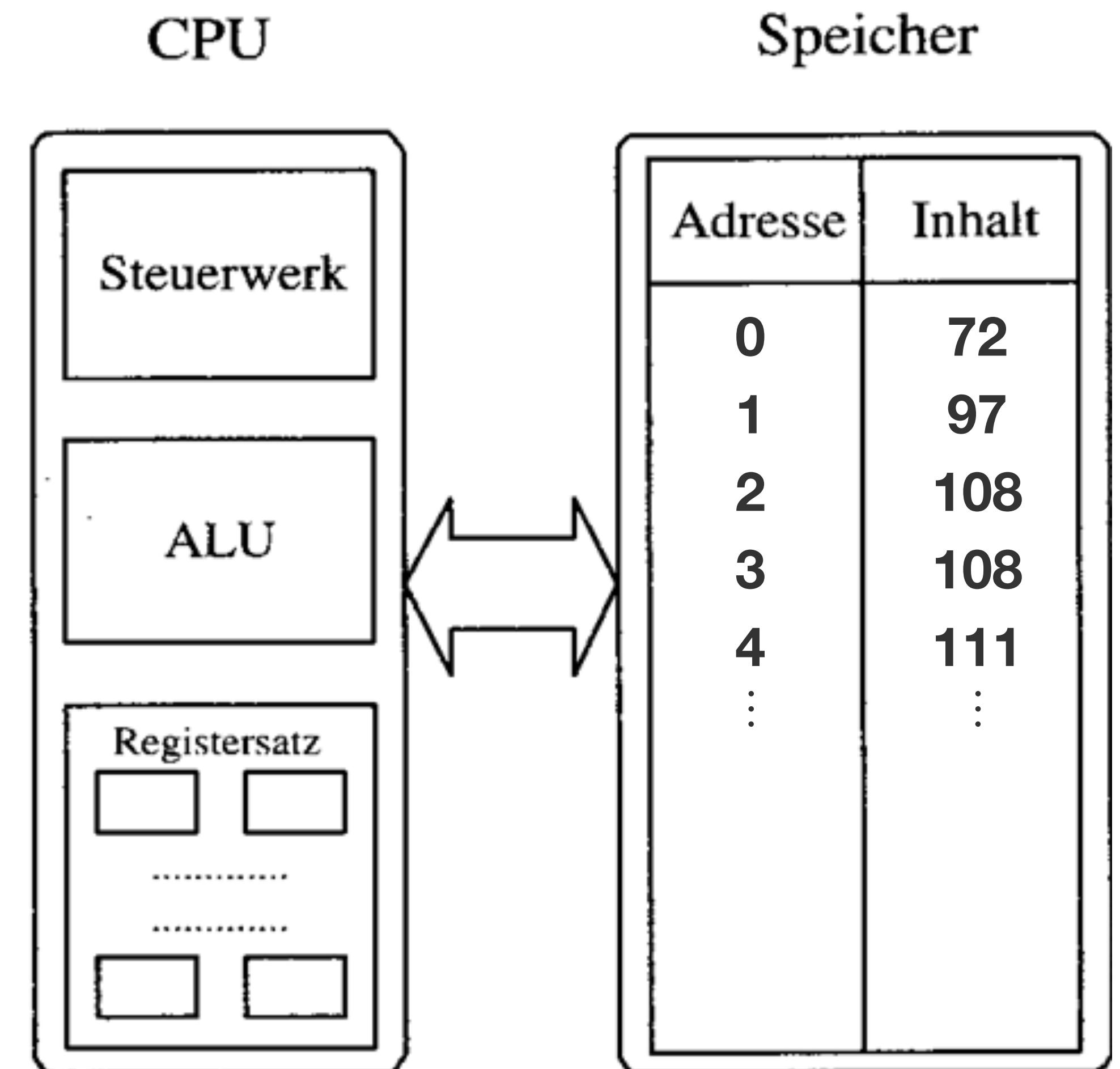
## von-Neumann-Architektur: Speicher

- Wahlfreier Zugriff (RAM)
- Jede Speicherzelle hat
  - eine Adresse
  - einen Inhalt
- Beispiel: ein PC mit 8 GB hat 8.589.934.592 Speicherzellen
- Programm und Daten im selben Speicher



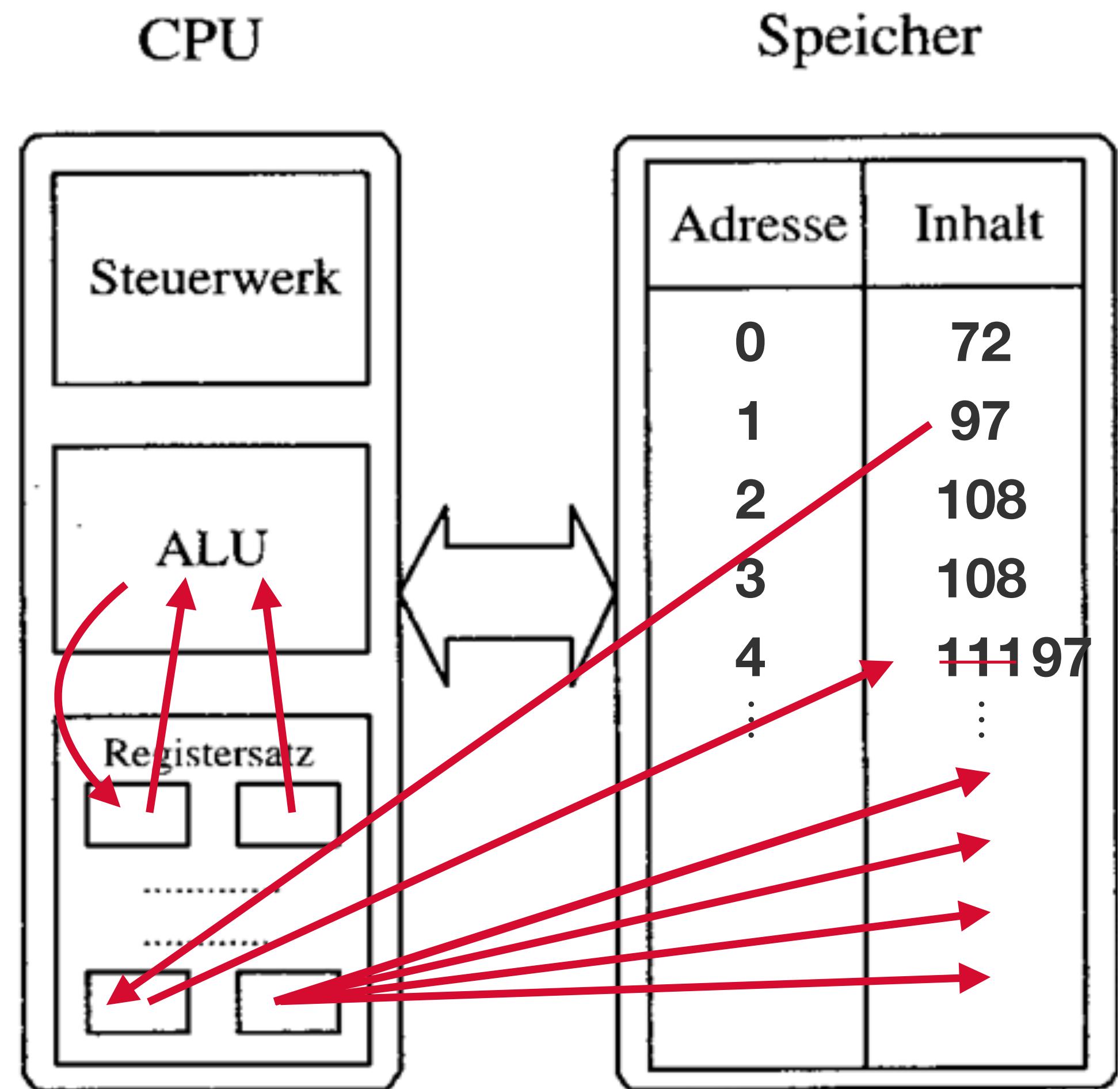
# Programmausführung

- Programme sind in Maschinensprache kodiert
- Prozessortyp-abhängig
  - z.B. **ARMv8** vs. **AMD64**
- Spezielles Register: Befehlszähler
- Fundamentaler Instruktions**zyklus** einer CPU
  - Befehl aus der Adresse im Befehlszähler lesen, Befehlszähler erhöhen, Befehl dekodieren, weitere Daten lesen, Befehl ausführen



# Programmausführung

- Typische Befehle
  - Laden (Speicher → Register)
  - Schreiben (Register → Speicher)
  - Verknüpfen von Registern (z.B. Addieren)
  - (bedingter) Sprung
- Takt
  - Befehle werden getaktet ausgeführt
  - Anzahl Takte/Befehl hängt vom Befehl ab



# Bits und Bytes

- 1 Bit kann 2 Zustände einnehmen: 0 oder 1
- Mehrere Bits:  $2^{\text{Anzahl}}$  Zustände
- Bei **n** Bits
  - Bit 0 = least significant bit (LSB)
  - Bit **n-1** = most significant bit (MSB)
- 1 Byte = 8 Bits
  - $2^8$  Zustände = 256 Zustände, d.h. 0 ... 255
  - Speicherzellen sind 1 Byte groß
- 1 Nibble = 4 Bits (Gut darstellbar durch eine Hexadezimalziffer 0..9, A-F)

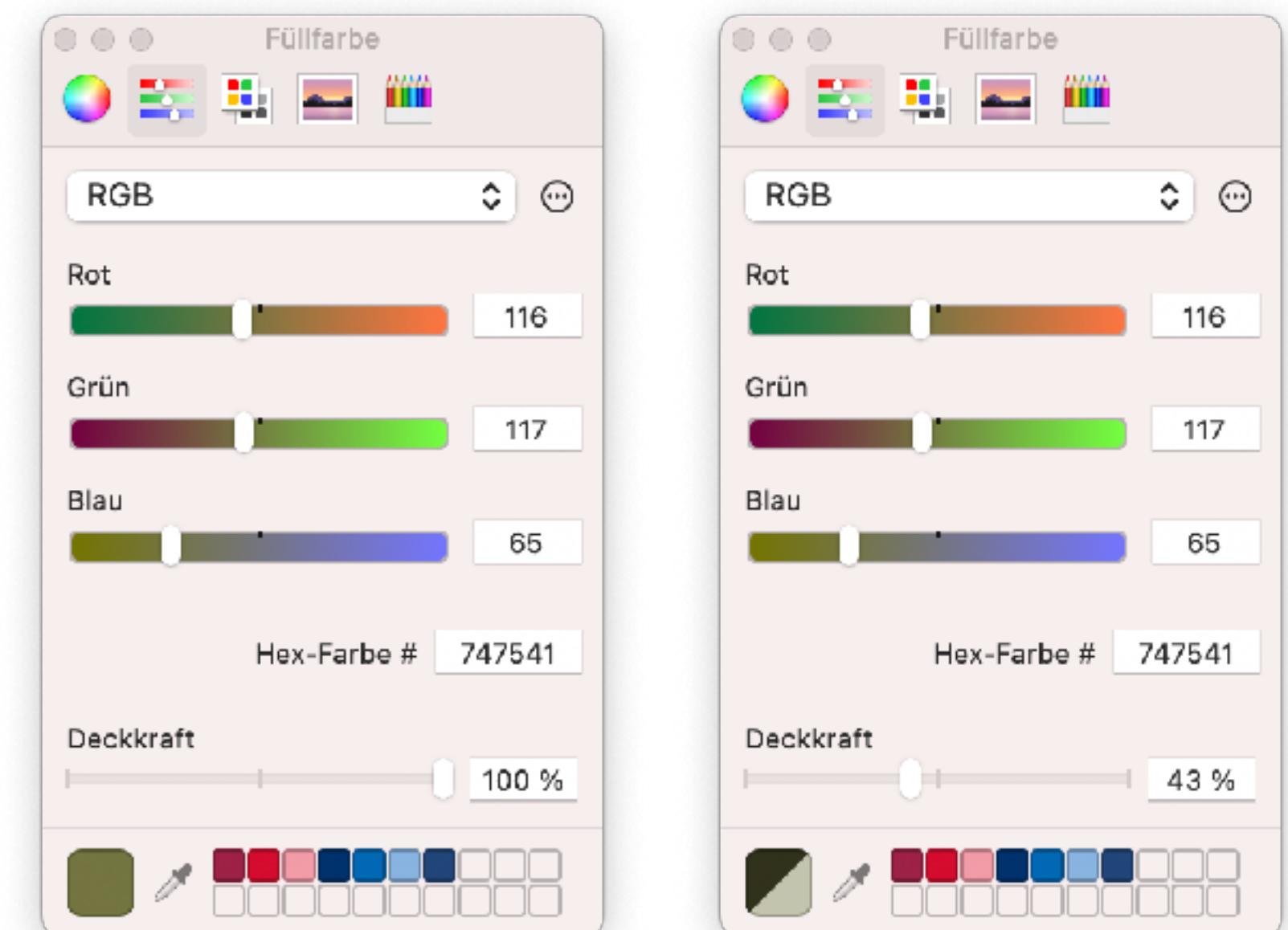
Bit 2	Bit 1	Bit 0	
0	0	0	Zustand 0
0	0	1	Zustand 1
0	1	0	Zustand 2
0	1	1	Zustand 3
1	0	0	Zustand 4
1	0	1	Zustand 5
1	1	0	Zustand 6
1	1	1	Zustand 7

## Größenangaben

- Üblich in der Informatik (oft ohne **i**)
  - Kilobyte, KiB:  $2^{10} = 1.024$  Bytes
  - Megabyte, MiB:  $2^{20} = 1.048.576$  Bytes
  - Gigabyte, GiB:  $2^{30} = 1.073.741.824$  Bytes
  - Terabyte, TiB:  $2^{40} = 1.099.511.627.776$  Bytes
- Im normierten Sprachgebrauch
  - Kilobyte, KB:  $10^3 = 1.000$  Bytes
  - Megabyte, MB:  $10^6 = 1.000.000$  Bytes
  - Gigabyte, GB:  $10^9 = 1.000.000.000$  Bytes
  - Terabyte, TB:  $10^{12} = 1.000.000.000.000$  Bytes

# Zustände und ihre Interpretationen

- Bits im Speicher repräsentieren Zustände, aber ihre Bedeutung hängt davon ab, wie sie interpretiert werden
- 4 Bytes (z.B. **65 117 116 111**) können z.B.
  - als ganze Zahl gedeutet werden (Integer, z.B. **1.869.903.169**)
  - als kurze Gleitkommazahl (Float, z.B. **7,56561 · 10<sup>28</sup>**)
  - als Folge von vier Zeichen aus jeweils einem Byte (String, z.B. „**Auto**“)
  - als Folge von Befehlen für den Prozessor
    - INC CX**
    - JNZ +116**
    - DB 111**
  - als Farbe (vom Grafikchip, z.B.  )



## Zeichensätze: ASCII

- American Standard Code for Information Interchange

- ISO 7 Bit

- Einheitlich in allen aktuellen Zeichensätzen

- Zeilenenden

- cr+lf (Windows)

- lf (Unix, macOS)

- cr (MacOS ≤ V9)

hex	0	10	20	30	40	50	60	70
0	nul	dle		0	@	P	'	p
1	soh	dc1	!	1	A	Q	a	q
2	stx	dc2	"	2	B	R	b	r
3	etx	dc3	#	3	C	S	c	s
4	eot	dc4	\$	4	D	T	d	t
5	enq	nak	%	5	E	U	e	u
6	ack	syn	&	6	F	V	f	v
7	bel	etb	,	7	G	W	g	w
8	bs	can	(	8	H	X	h	x
9	ht	em	)	9	I	Y	i	y
A	lf	sub	*	:	J	Z	j	z
B	vt	esc	+	;	K	[	k	{
C	ff	fs	,	<	L	\	l	
D	cr	gs	-	=	M	]	m	}
E	so	rs	.	>	N	^	n	~
F	si	us	/	?	O	-	o	del

## Latin-1 / Latin-9

- ISO 8859-1
- ISO 8859-15
- Standardkodierung unter Windows für 8-Bit-Zeichen

	...0	...1	...2	...3	...4	...5	...6	...7	...8	...9	...A	...B	...C	...D	...E	...F
0...	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1...	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2...	SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3...	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4...	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5...	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	-
6...	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7...	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
8...	PAD	HOP	BPH	NBH	IND	NEL	SSA	ESA	HTS	HTJ	VTS	PLD	PLU	RI	SS2	SS3
9...	DCS	PU1	PU2	STS	CCH	MW	SPA	EPA	SOS	SGCI	SCI	CSI	ST	OSC	PM	APC
A...	NBSP	í	ø	£	¤ €	¥	: š	§	„ š	©	ª	«	¬	SHY-	®	-
B...	°	±	²	³	‘ ž	µ	¶	·	„ ž	¹	º	»	¼ œ	½ œ	¾ Ÿ	¿
C...	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D...	Đ	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E...	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F...	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

# Unicode

- 32-Bit-Zeichensatz
- Ordnet Zeichen nach Alphabeten
- Unicode 14.0: 144.697 Zeichen
- UTF-16
- Java
- Windows

	000	001	002	003	004	005	006	007
0	NUL	DLE	SP	0	@	P	`	p
1	SCH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

	2F8C	2F8D	2F8E	2F8F	2F90	2F91	2F92	2F93	2F94	2F95	2F96
0	擎	晤	枅	欸	派	滬	饗	瑱	直	碩	笄
1	掩	暉	衆	歎	海	漘	爵	璫	昞	礪	簷
2	攬	冒	梅	歡	流	瀆	暉	瓊	昇	枚	篆
3	摩	冕	柵	歲	浩	淪	醑	瓶	眎	租	築
4	強	最	枹	殮	浸	滌	犀	牲	眾	祓	篤
5	撝	普	栟	殺	涅	瀛	犧	串	眞	禡	纂
6	搃	肭	檣	蔻	滌	瀶	犴	眰	眞	福	糒
7	攢	耽	櫛	殿	洴	瀶	狽	眱	眞	秋	𩙴
8	敏	朗	楂	莓	港	災	獺	異	昧	秆	熳
9	敬	望	橒	旛	涇	卉	王	眵	眥	穀	穠
A	敷	腰	概	汎	澁	炭	珎	眵	眴	紀	𦨑
B	既	杞	様	沈	滋	薰	玥	曉	眴	穏	糲
C	書	杓	樞	滇	燄	堡	𦨑	𦨑	否	𦨑	𦨑
D	晉	束	櫛	涙	燦	壘	盈	𦨑	𦨑	𦨑	𦨑
E	曉	朮	櫟	汎	淹	燔	𦨑	𦨑	𦨑	𦨑	𦨑
F	暑	朂	次	溟	潮	美	瑜	𦨑	𦨑	𦨑	𦨑

## UTF-8

- Kompakte Kodierung von Unicode für Dateien
  - Zeichen werden durch 1 bis 4 Bytes kodiert
    - Dateilänge hängt nicht nur davon ab, wie viele Zeichen in Datei enthalten sind, sondern auch welche
  - Am Dateianfang kann eine BOM stehen (Byte Order Mark)
    - Für LaTeX darf keine BOM in der Datei stehen!
    - Standard bei Linux, macOS und PI-1
- | Unicode-Bereich     | UTF-8-Kodierung                     |
|---------------------|-------------------------------------|
| 0000 0000–0000 007F | 0xxxxxxx                            |
| 0000 0080–0000 07FF | 110xxxxx 10xxxxxx                   |
| 0000 0800–0000 FFFF | 1110xxxx 10xxxxxx 10xxxxxx          |
| 0001 0000–0010 FFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx |

## Nächste Schritte

- Praktikum wählen, falls noch nicht geschehen
  - Ohne Zuordnung zu Praktikum im Stud.IP keine Abgabe möglich
  - Ankündigungen der Tutor:in gehen an einem vorbei (E-Mail lesen!)
- Praktika beginnen nächste Woche
  - In Online-Praktika kann jede Tutor:in ihr eigenes Kommunikationswerkzeug wählen
- Installationen: TeX, BlueJ und PI-1-Vorlagen installieren