

PROGRAMMIEREN I

WS 2022

Prof. Dr. Kolja Eger
Hochschule für Angewandte Wissenschaften Hamburg

Check.In

- Intro: <https://www.youtube.com/watch?v=WrSi3sCVGck>
- Intro 2: <https://www.youtube.com/watch?v=IY7EsTnUSxY>

Vorstellung

#open-mind

#open-heart

#open-door

- Seit WS21 an der HAW
- Seit WS22 Studiengangsleiter des REE
- Mentor im REE für 1.Semester
- Professor für „IT für verteilte Energiesysteme“
- Über 13 Jahre IT-Erfahrung im Energiebereich (E.ON & Siemens)
- Promotion an der TU Hamburg-Harburg
- Studium der Informations- und Kommunikationstechnik

Prof. Dr. Kolja Eger



Kontakt:

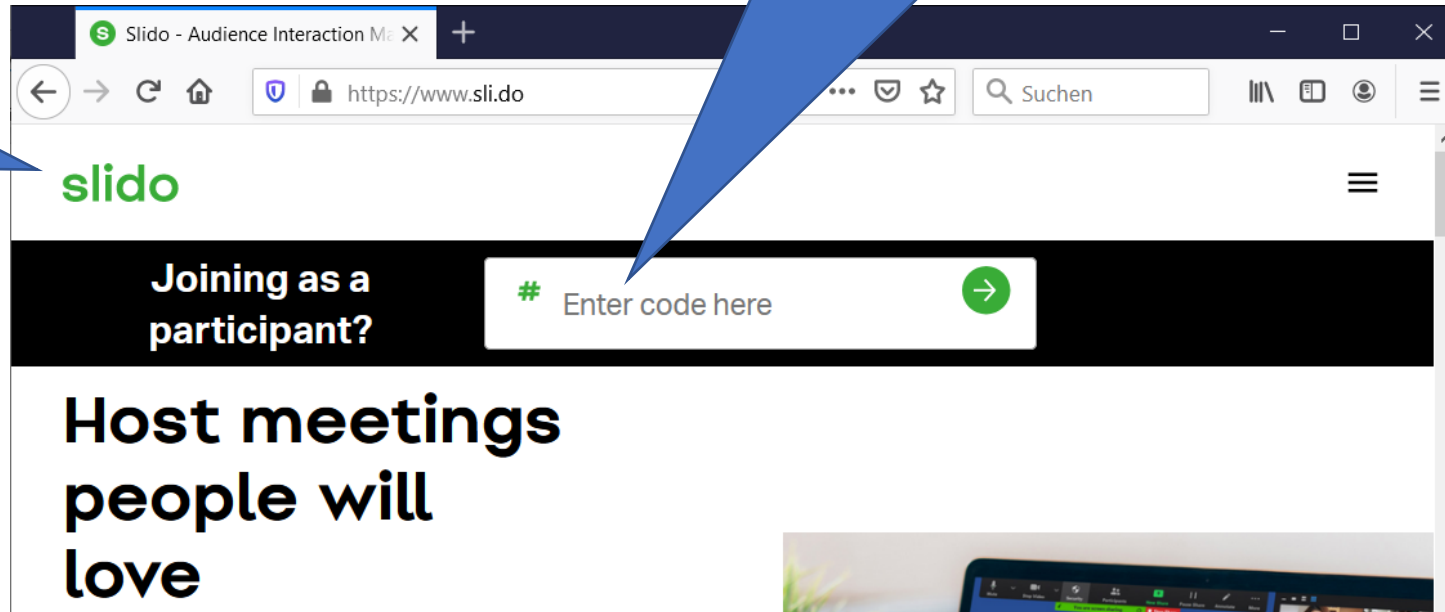
E-Mail kolja.eger@haw-hamburg.de

Büro 12.80

Anruf über MS Teams


Slido.com

REE-PR1



Programmiersprachen

- Welche Programmiersprache?
 - Hauptsache programmieren!
 - Konzepte verstehen
 - Strukturiert & lösungs-orientiert Denken
 - Wir lernen C
 - Wer noch was anderes ausprobieren möchte, hier mal reingucken: <https://sumo.blogs.uni-hamburg.de/wp-content/uploads/lernmodule/lernen/programmieren/which-programming-language-should-i-learn-first-infographic.pdf>
- Popularität von Programmiersprachen
 - Tiobe Index <https://www.tiobe.com/tiobe-index/>
 - Statista/PYPL <https://de.statista.com/infografik/16544/anteile-der-populaersten-programmiersprachen-weltweit/> & <https://pypl.github.io/PYPL.html>



Was kann ich
am Ende des
Semesters?

Learning Outcome für PR1 – Programmieren 1, 1.Semester, Informations- und Elektrotechnik (Kolja Eger)

Wer	Die Studierenden..
Was	.. können Programme in C implementieren und testen und Problemstellungen mithilfe der C-Programme lösen/berechnen
Womit	Indem sie unterschiedlichen Datentypen sowie Arrays & Zeiger nutzen, Ein- und Ausgabe auf der Kommandozeile erstellen und Dateien ein-/auslesen, Schleifen und Anweisungen programmieren, und mit Funktionen , Headerdateien, Makros Programme und dynamischen Speicher aufbauen und die Entwicklungsumgebung Visual Studio (inkl. Debugger) unter Windows bedienen
Wozu	Um ingenieurhafte Probleme/Aufgaben mit Software-Programmen lösen zu können

Präsenzklausur im Labor (180min)

Dynamische Speicherallokation

Praktikum 7

Strukturen

Praktikum 6

Dateien

Praktikum 5

Zeiger

Praktikum 4

Vektoren (Arrays)

Praktikum 3

Funktionen

Praktikum 2

Kontrollstrukturen

Datentypen & Operatoren

Praktikum 1

Erste Programme

Unser Weg durch das Semester

Wie wird in PR1 gearbeitet?

Vorlesung → seminaristischer Unterricht

- Live Coding
- Programmier-Übungsaufgaben
- Umfragen/Quiz
- Diskussion
- Präsentation/„Frontalbeschallung“
- ..

→ Bringen Sie Ihren Laptop mit und programmieren Sie parallel mit!

Praktikum → Prüfungsvorleistung (PVL)

- 7 Termine & 7 Aufgaben
- 3 Gruppen mit kleinerer Gruppengröße
- Anwesenheit & Abnahme sind Pflicht
- Vorbereitung notwendig!
- ..

→ Später mehr..

Unterlagen & Tools

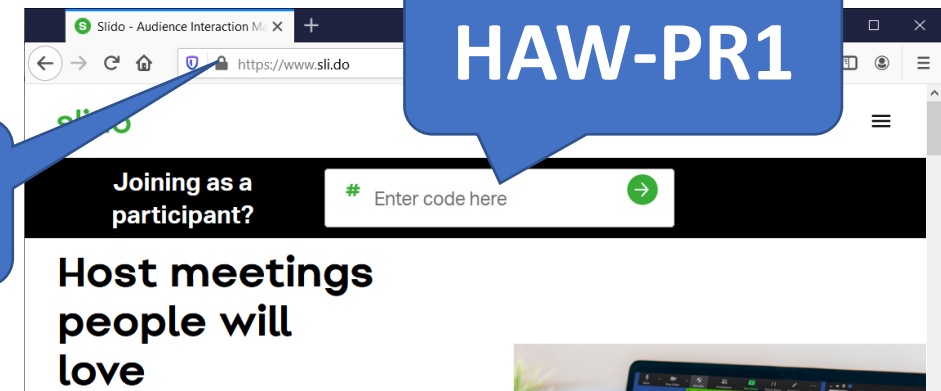
- Vorlesung baut auf den Skripten Programmieren-1 und Programmieren-2 von Prof. Heß auf → <http://www.rrhess.de/>
- Geht nicht auf alle Inhalte der Skripte ein
- Slides der Vorlesung & Praktikumsaufgaben in EMIL
- EMIL
 - **Einschreibeschlüssel: #HAW-PR1**
- Sli.do → Umfragen
- **Fragen jederzeit willkommen!**

EMIL: Programmieren 1 W22



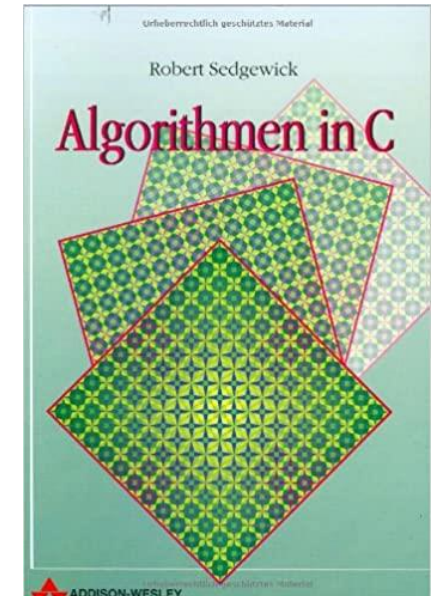
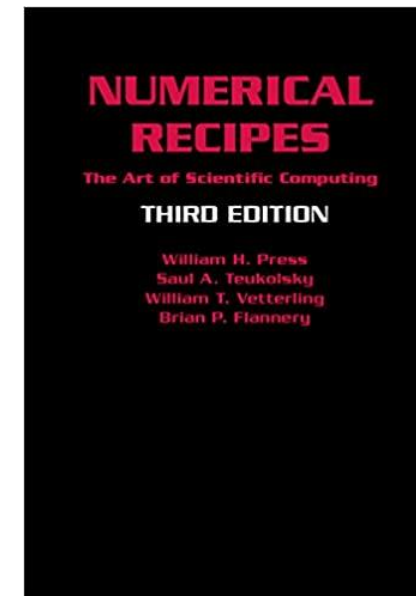
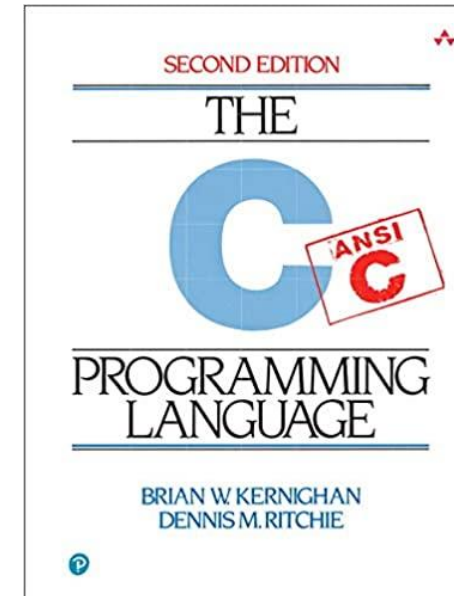
Slido.com

HAW-PR1



Weitere Quellen

- Online Hilfe von Visual Studio
 - <https://docs.microsoft.com/de-de/visualstudio/get-started/visual-studio-ide?view=vs-2022&viewFallbackFrom=msvc-170>
- C-Sprachreferenz, z.B. <https://cplusplus.com/reference/clibrary/>
- Bücher →
- Google & YouTube
 - Aber Vorsicht – C hat viele Möglichkeiten, die wir nicht alle in der Veranstaltung kennenlernen werden. Fokussieren Sie sich auf die Wege, die wir hier durchnehmen!



Tutorium

- Tutorium Offener Lernraum für Programmieren 1&2/OOP Veranstaltungen im ersten Studienjahr ALLER Studiengänge
- Teams-Raum: <https://teams.microsoft.com/l/team/19%3AYmiysQFSUWVRIHkU0aTdH0gGszPu6Kg7XZuJtPQ-40o1%40thread.tacv2/conversations?groupId=1a7fa425-74e0-4dd4-a1e7-8457364c75c0&tenantId=38d63075-6a27-4ec4-95f9-473f5ef2f1b5>

Entwicklungsumgebung

(IDE - Integrated Development Environment)

- Software zum Schreiben, Übersetzen und Managen von Quellcode
- Verschiedene Tools für C verfügbar
 - sowohl Open Source als auch kommerziell
 - Windows / Linux
- Wir verwenden zurzeit Visual Studio 2019 von Microsoft (unter Windows)



NetBeans



Visual Studio

...

Wo bekomme ich für meinen Laptop
Visual Studio 2019 Enterprise her?

Schritt 1: Über HAW Online Services Zugang zu Microsoft Azure Education erhalten

The screenshot shows the HAW Hamburg website. The top navigation bar is blue with the HAW Hamburg logo on the left and links for BESCHÄFTIGTENPORTAL, QUICKLINKS, and ENGLISH on the right. Below this is a secondary navigation bar with links for HOCHSCHULE, STUDIUM, FORSCHUNG, INTERNATIONAL, and DIGITALISIERUNG. A banner image below the navigation bar shows chess pieces and a person in a white lab coat. Below the banner, a breadcrumb trail reads: STARTSEITE — QUICKLINKS — ONLINE-SERVICES — KOSTENLOSE SOFTWARE — MICROSOFT AZURE DEV TOOLS FOR TEACHING. On the left, a blue sidebar contains a list of links under the heading 'Kostenlose Software'. The link 'Microsoft Azure Dev Tools for Teaching' is highlighted. The main content area has the title 'Microsoft Azure Dev Tools for Teaching' in large blue font. Below the title, a paragraph explains that the program (formerly Microsoft DreamSpark or Microsoft Imagine) provides students, teachers, and employees of HAW Hamburg with access to a large selection of Microsoft products for free use in research and teaching. A list of three steps follows: 1. Register with your HAW email address for Microsoft Office 365 (if you haven't used Azure Dev Tools for Teaching, DreamSpark, or Imagine before, you need to link a Microsoft business account to your HAW email address). 2. You will receive a confirmation code and can complete the registration. 3. Download the desired programs after logging in with your account. At the bottom, there is a link 'Wann und wo darf die Software verwendet werden?' followed by a plus icon in a circle.

HAW HAMBURG

BESCHÄFTIGTENPORTAL QUICKLINKS ENGLISH

HOCHSCHULE STUDIUM FORSCHUNG INTERNATIONAL DIGITALISIERUNG

STARTSEITE — QUICKLINKS — ONLINE-SERVICES — KOSTENLOSE SOFTWARE — MICROSOFT AZURE DEV TOOLS FOR TEACHING

« < Online-Services

Kostenlose Software

- Microsoft Office 365
- Microsoft Azure Dev Tools for Teaching**
- Webkonferenz-Tools
- SOPHOS Antivirus-Programm

Microsoft Azure Dev Tools for Teaching

Das „Microsoft Azure Dev Tools for Teaching“ Programm (früher "Microsoft DreamSpark" oder "Microsoft Imagine") bietet Studierenden, Lehrenden und Beschäftigten der HAW Hamburg die Möglichkeit, eine große Auswahl von Microsoft-Produkten **im Bereich Forschung und Lehre kostenlos zu nutzen**.

1. Registrieren Sie sich mit Ihrer HAW-Mail-Adresse für [Microsoft Office 365](#)
(wenn Sie bisher noch nie "Azure Dev Tools for Teaching" oder "Dreamspark" oder "Imagine" genutzt und daher noch kein Microsoft-Geschäftskonto mit der HAW-Mail-Adresse hinterlegt haben)
2. Sie erhalten einen Bestätigungscode und können die Registrierung abschließen.
3. Laden Sie die gewünschten Programme herunter, nachdem Sie sich [mit Ihrem Konto eingeloggt haben](#)

Wann und wo darf die Software verwendet werden? (+)

<https://www.haw-hamburg.de/online-services/kostenlose-software/microsoft-azure-dev-tools-for-teaching/>

Schritt 2: Product Key für Visual Studio Enterprise 2019 herunterladen

Microsoft Azure

Nach Ressourcen, Diensten und Dokumenten suchen (G+/)

Kolja.Eger@haw-hambu...
HAW-HAMBURG.DE (HAWHAMB...)

Home > Education

Education | Software

Overview

Get started

Learning resources

Roles

Software

Learning

My account

Profile

Need help?

Support

« visual

Product category : All

Operating System : All

System type : 64 bit

Product language : German,Multilanguage,English

10 Items

Name ↑↓	Product category ↑↓	Operating System ↑↓	System type ↑↓
Visual Studio Enterprise 2019	Developer Tools	Windows	64 bit
Visual Studio Enterprise Edition 2022	Developer Tools	Windows	64 bit
Agents for Visual Studio 2019 (version 16.0) Test Ag...	Developer Tools	Windows	64 bit
Agents for Visual Studio 2019 (version 16.0) Test Co...	Developer Tools	Windows	64 bit
Remote Tools for Visual Studio 2019 (version 16.0)	Developer Tools	Windows	64 bit
Remote Tools for Visual Studio 2019 (version 16.0)	Developer Tools	Windows	64 bit
Visual Studio 2019 for Mac	Developer Tools	Mac	64 bit
Visual Studio Code	Developer Tools	Windows	64 bit
Visual Studio Community 2019 (version 16.0)	Developer Tools	Windows	64 bit
Visual Studio für Mac	Developer Tools	Mac	64 bit

Software

Education

Visual Studio Enterprise 2019

An integrated, end-to-end solution for developers looking for high productivity and seamless coordination across teams of any size.

Operating System
Windows

Product language
Multilanguage

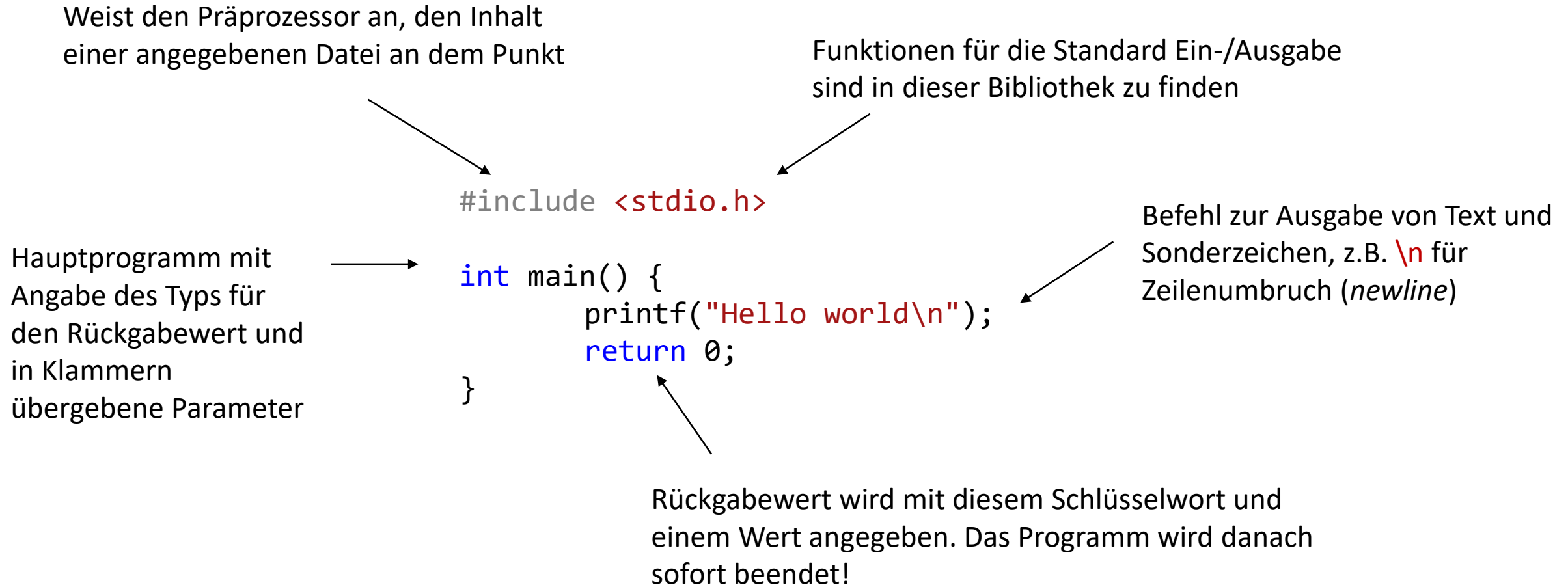
System
64 bit

View Key

Download Cancel

UND JETZT GEHT ES LOS..

→ Beispiel in Visual Studio



Variablen

- In einem Computerprogramm werden Daten gelesen, verarbeitet und ausgegeben
- Daten werden hierfür in Variablen gespeichert
- Zuweisung in C mit

```
Wurzel2 = 1.4142;
```

Name der Variablen

Zugewiesener Wert

- Zugewiesener Wert ist entweder eine Konstante oder eine Variable oder ein Ausdruck
- Beispiel für Ausdruck:

```
Celsius = Kelvin-273;
```

Operator

Variable

Konstante

Variablen (II)

- Beispiel für verschachtelten Ausdruck:

```
Celsius = 5 * (Fahr - 32) / 9;
```

- und allgemein
 - Zuweisung: *Variable = Ausdruck/Variable/Konstante*
 - Ausdruck: *Ausdruck/Variable/Konstante **Operator** Ausdruck/Variable/Konstante*
- In C haben Variablen einen Typ (z.B. Ganzzahl, Gleitkommazahl, Buchstabe)
- Bevor eine Variable verwendet wird, muss sie definiert werden, z.B.

```
int Anzahl;
```

```
double Celsius;
```

Eine Variable hat 4 Eigenschaften

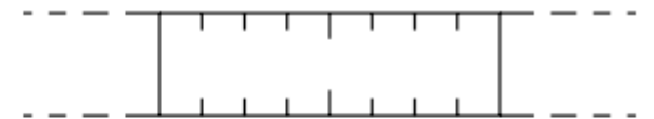
- Typ
 - definierter Datentyp, z.B. `int`, `char`, ..
- Name
 - um Variablen von anderen zu unterscheiden
- Adresse/Speicher
 - eine Variable wird an einer Stelle im Speicher abgelegt, die durch eine eindeutige Adresse definiert ist.
- Wert
 - eine Variable hat einen Wert

```
double zahl = 1.2345;
```

Typ: double

Name: zahl

Speicher:



Wert: 1,234

→ Beispiel in Visual Studio

Kommentare helfen den Code besser zu verstehen!

Zwei Formen:

```
/* Kommentar  
.. 2.Zeile des Kommentars  
*/
```

Oder auch (ab C-Version C99)

```
// Einzeiliger Kommentar
```

Definition der Variable
mit Namen ‚a‘ und Typ int
(Ganzzahl)

```
#include <stdio.h>
```

```
/* Verwendung von Variablen */
```

```
int main()
```

```
{
```

```
int a;
```

```
*/
```

/* Definition einer Variable

C kennt viele Operatoren,
u.a. Grundrechenarten
(+, -, *, /) oder
Restdivision (%)

```
a = 7;
```

```
a = a + 4;
```

```
Operators */
```

/* Zuweisung eines Werts */

/* Verwendung eines

```
printf("Ergebnis: %d\n", a);
```

```
return 0;
```

```
}
```

Platzhalter für die Ausgabe
von Variablen, %d für int

Definition von Variablen

Allgemein

```
Datentyp Variablenname [, Variablenname [... ]];
```

- Optionale Angaben in eckigen Klammern []
- Es können mehrere Variablen gleichzeitig definiert werden
- Variablenname darf nur einmal innerhalb eines Blocks definiert werden (Datentypen dürfen mehrfach genutzt werden)
- Einer Variablen kann bei der Vereinbarung ein Startwert zugewiesen werden.
- Die vorgestellte Bezeichnung `const` gibt an, dass die Variable nicht verändert werden kann
 - meistens Compiler-Fehler (auch in Visual Studio)
 - (allgemein: undefinierte Folgen)

Beispiele:

```
int a, b;
```

```
int counter = 0;
```

```
const float pi = 3.141;
```

Variablennamen

- Ein Variablenname besteht aus großen und kleinen Buchstaben, Ziffern und dem Tiefstrich “_”
- Die deutschen Umlaute sind nicht erlaubt
- Erste Zeichen ist ein Buchstabe oder ein Tiefstrich (keine Ziffer)
- Große und kleine Buchstaben werden unterschieden
- Name darf beliebig lang sein
- Variablennamen werden nur anhand der ersten 31 Zeichen unterschieden
- Reservierte Wörter wie *if*, *while* oder *break* dürfen nicht verwendet werden

Elementare Datentypen

Typ	Beschreibung
char	ganzzahliger Wert (ein Byte), bzw. ein Zeichen/Buchstabe (engl. <i>character</i>)
int	ganzzahliger Wert in der auf dem Rechner 'natürlichen' Größe
float	Gleitkommazahl mit einfacher Genauigkeit
double	Gleitkommazahl mit doppelter Genauigkeit

Wertebereiche von Datentypen

- Wertebereiche der Variablen hängen von der Umgebung und dem entsprechenden Compiler ab
- Variablen benötigen Speicher und abhängig wie viel Speicher sie belegen, ist der Wertebereich ausgelegt
- Größe eines Datentyps kann mit der Funktion `sizeof()` ermittelt werden, z.B. Ausgabe mit

```
printf("%d", sizeof(int));
```

- `sizeof()` liefert die Größe in Bytes (und 1 Byte = 8 Bits)
- Wertebereich für Ganzzahlen ohne Vorzeichen (mit N = Anzahl der Bits)
 - Mit Vorzeichen: $-2^{(N-1)}$ bis $+2^{(N-1)}-1$
 - Ohne Vorzeichen: Von 0 bis $2^N - 1$
- Wertebereiche für Compiler in Visual Studio:
 - Stehen in der Include-Datei „limits.h“
 - Oder <https://docs.microsoft.com/de-de/cpp/cpp/data-type-ranges?view=msvc-170>

Übung: Wertebereich für `int`

Arduino ist ein kleiner Computer für unterschiedlichste Projekte. Ideal zum Lernen von C und Steuerung von I/Os 😊



Wertebereich: $-2^{(N-1)}$ bis $+2^{(N-1)} - 1$

- `int` in Visual Studio → 4 Byte
- `int` auf Arduino Uno → 2 Byte
- Wie groß sind die Wertebereiche für beide Umgebungen?
- Diskutieren Sie mit Ihrem Nachbarn!

Praktikum

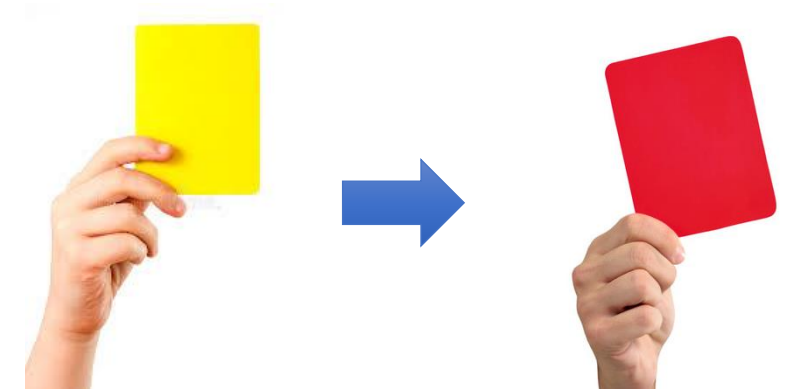
- Praktikum ist eine Prüfungsvorleistung – nur wenn Sie das Praktikum erfolgreich durchlaufen, werden Sie zur Klausur zugelassen
- 7 Praktikumsaufgaben pro Gruppe → ab übernächster Woche geht es los!
- 3 Gruppen → 2 Gruppen am Donnerstag und 1 Gruppe am Freitag
- Nutzen Sie auch die Rechner im Labor, um die **Umgebung kennenzulernen**, auf der Sie auch die **Klausur** schreiben werden → Visual Studio in Englisch!!!
- Sie müssen sich für eine Praktikumsgruppe anmelden → stisys

Praktikum (7 Termine pro Gruppe)

- Zusammen mit Herrn Nieder betreue ich das Praktikum
- Generell arbeiten Sie in **2er-Teams** (in Ausnahme 3er-Teams) → Teamfindung im ersten Praktikumstermin
- Ein Testat für ein Praktikum erhalten Sie **nach Demonstration Ihrer Lösung während des Praktikumstermins** inkl. Erläuterungen & Beantwortung von Fragen
- Bei Krankheit (mit Vorlage eines Attest) muss der Termin nachgeholt werden
- Nach erfolgreicher Demonstration können Sie die Zeit im Praktikumstermin nutzen um den nächsten Termin vorzubereiten.
- Ihre **Anwesenheitspflicht** ist **nach der erfolgreichen Abnahme aufgehoben** → gute Vorbereitung zahlt sich aus!

Praktikum - Spielregeln

- Grundsätzlich müssen Sie die Praktikumsaufgabe **vorbereiten, im Termin fertigstellen, vorstellen und sich abnehmen lassen.**
- **Jeder muss alle Lösungen präsentieren können**
- Falls Sie dies zeitlich nicht schaffen, kann eine Abnahme noch **bis zu 7 Tage später** erfolgen (Terminabsprache muss von Ihnen initiiert werden!)
- Falls Sie verspätet bzw. keine Lösung einreichen oder ihre Lösung qualitativ mangelhaft ist, ist diese Praktikumsaufgabe nicht bestanden
 - Sie erhalten dann eine **Verwarnung** („gelbe Karte“)
 - Bei der zweiten Verwarnung, erhalten Sie **kein Testat** („gelb-rote Karte“).
- Als **Prüfungsvorleistung** müssen Sie **alle Praktika durchführen** (und höchstens eine Verwarnung erhalten)



Praktikumsaufgabe 1 in EMIL

Ganzzahlige Datentypen

- Elementare ganzzahlige Datentypen sind `int` und `char`
- `char` kann ein Zeichen/Buchstaben speichern bzw. stellvertretend eine kleine Zahl
- Neben `int` können auch kleinere/größere Ganzzahl definiert werden:
 - Kleinere (min. 2 Bytes): `short int` (oder in kurz) `short`
 - Größere (min. 4 Bytes): `long int` bzw. `long`
- Den Datentypen `char`, `short`, `int` und `long` kann das Wort `signed` oder `unsigned` vorgestellt werden
- Dies legt fest, ob sich der Wertebereich auf positive & negative oder nur auf positive Zahlen beschränkt
- `short`, `int` und `long` sind ohne Angabe vorzeichenbehaftet (`char` nicht)
- Ist nur `signed` oder `unsigned` angegeben, handelt es sich um den Datentyp `int`

Ganzzahlige Datentypen - Beispiele

```
3  int main() {  
4      short a;      // short integer (mit Vorzeichen)  
5      unsigned b;    // positiver int  
6      unsigned short c; // short integer (ohne Vorzeichen)  
7      signed long d;  // long integer (mit Vorzeichen)  
8      signed char e;  // char mit Vorzeichen (Wertebereich: -128 bis 127)  
9  }
```

Wie werden Ganzzahlen interpretiert?

Beispiele:

- Standardmäßig werden alle Zahlen als `int` interpretiert
- Aber die Angabe in anderen Datentypen ist auch möglich, z.B.
 - Typ `long` wird durch „l“ oder „L“ erzeugt
 - Vorzeichenlose Zahl: „u“ oder „U“
 - Kombination möglich „ul“
- Auch die Eingabe in anderen Zahlensystemen möglich → Dazu in einer anderen Vorlesung mehr!
 - Hexadezimalzahl mit „0x“ oder „0X“
 - Oktalzahl beginnt mit einer Null
- Kombination möglich

1234

1234L

1234u

1234ul

0x3f

(ergibt dezimal: 63)

045

(ergibt dezimal: 37)

0xFUL

(dezimal 15 vom Typ
unsigned long)

Datentyp `char`

- Steht sowohl für eine kleine Zahl als auch für ein Zeichen
- D.h. einer Variablen vom Typ `char` können Sie ein Zeichen zuweisen (einfache Anführungszeichen!)
- Der numerische Wert ist im ASCII-Code festgelegt
 - ASCII = American Standard Code for Information Interchange
 - Definition von 128 Zeichen (Code 0 bis 127) (siehe nä. Slide)
- C unterscheidet nicht zwischen Zeichen und Ganzzahlen, da Zeichen intern als Ganzzahlen dargestellt werden
- Sie können mit Zeichen rechnen!
- Sonderzeichen wird ein `\` vorangestellt, z.B. „`\n`“ für Zeilenvorschub

```
char a = 'B';
```

ASCII-Code von 'B' ist 66

```
char a = 'A' + 1;
```

ASCII-Tabelle

Sonderzeichen,
z.B. ESC = Escape,

Dez/Hex/Okt	Zeichen	Dez/Hex/Okt	Zeichen	Dez/Hex/Okt	Zeichen	Dez/Hex/Okt	Zeichen
0/00/000	NUL	32/20/040	SP	64/40/100	@	96/60/140	`
1/01/001	SOH	33/21/041	!	65/41/101	A	97/61/141	a
2/02/002	STX	34/22/042	"	66/42/102	B	98/62/142	b
3/03/003	ETX	35/23/043	#	67/43/103	C	99/63/143	c
4/04/004	EOT	36/24/044	\$	68/44/104	D	100/64/144	d
5/05/005	ENQ	37/25/045	%	69/45/105	E	101/65/145	e
6/06/006	ACK	38/26/046	&	70/46/106	F	102/66/146	f
7/07/007	BEL	39/27/047	'	71/47/107	G	103/67/147	g
8/08/010	BS	40/28/050	(72/48/110	H	104/68/150	h
9/09/011	TAB	41/29/051)	73/49/111	I	105/69/151	i
10/0A/012	LF	42/2A/052	*	74/4A/112	J	106/6A/152	j
11/0B/013	VT	43/2B/053	+	75/4B/113	K	107/6B/153	k
12/0C/014	FF	44/2C/054	,	76/4C/114	L	108/6C/154	l
13/0D/015	CR	45/2D/055	-	77/4D/115	M	109/6D/155	m
14/0E/016	SO	46/2E/056	.	78/4E/116	N	110/6E/156	n
15/0F/017	SI	47/2F/057	/	79/4F/117	O	111/6F/157	o
16/10/020	DLE	48/30/060	0	80/50/120	P	112/70/160	p
17/11/021	DC1	49/31/061	1	81/51/121	Q	113/71/161	q
18/12/022	DC2	50/32/062	2	82/52/122	R	114/72/162	r
19/13/023	DC3	51/33/063	3	83/53/123	S	115/73/163	s
20/14/024	DC4	52/34/064	4	84/54/124	T	116/74/164	t
21/15/025	NAK	53/35/065	5	85/55/125	U	117/75/165	u
22/16/026	SYN	54/36/066	6	86/56/126	V	118/76/166	v
23/17/027	ETB	55/37/067	7	87/57/127	W	119/77/167	w
24/18/030	CAN	56/38/070	8	88/58/130	X	120/78/170	x
25/19/031	EM	57/39/071	9	89/59/131	Y	121/79/171	y
26/1A/032	SUB	58/3A/072	:	90/5A/132	Z	122/7A/172	z
27/1B/033	ESC	59/3B/073	;	91/5B/133	[123/7B/173	{
28/1C/034	FS	60/3C/074	<	92/5C/134	\	124/7C/174	
29/1D/035	GS	61/3D/075	=	93/5D/135]	125/7D/175	}
30/1E/036	RS	62/3E/076	>	94/5E/136	^	126/7E/176	~
31/1F/037	US	63/3F/077	?	95/5F/137	_	127/7F/177	DEL

Groß- &
Kleinbuchstaben
(keine Umlaute)

Sonderzeichen im Überblick

<code>\n</code> Zeilenvorschub	<code>\r</code> Wagenrücklauf
<code>\t</code> Tabulator	<code>\v</code> Vertikaltabulator
<code>\b</code> backspace	<code>\f</code> Seitenvorschub
<code>\'</code> Einfachanführungszeichen	<code>\"</code> Doppelanführungszeichen
<code>\a</code> Klingelzeichen	<code>\?</code> Fragezeichen
<code>\ooo</code> ASCII-Code (oktal)	<code>\xhh</code> ASCII-Code (hexadezimal)
<code>\\</code> Gegenstrich	

Zeichenkette (Strings)

- Eine Zeichenkette wird durch doppelte Anführungszeichen angedeutet und kann beliebige Anzahl von Zeichen enthalten
- Es gelten die gleichen Sonderzeichen
- Zeichenketten können auch leer sein
- Aufeinanderfolgende Zeichenketten werden als eine interpretiert und können so auf mehrere Zeilen aufgeteilt werden
- Eine Zeichenkette ist ein Vektor von Zeichen → Vektoren werden wir noch in einer anderen Vorlesung behandeln
- Intern wird eine Zeichenkette durch null (`\0`) begrenzt. Deshalb benötigt eine Zeichenkette im Speicher ein Byte mehr als die Anzahl der Zeichen
- Beachte den Unterschied:
 - `'x'` → ein Zeichen
 - `"x"` → eine Zeichenkette mit null-Terminierung

Zeichenkette

Sonderzeichen

```
printf("Hello world\n");
```

```
printf("");
```

```
printf("Hello "  
      "World" "\n");
```

→ Beispiel in Visual Studio

```
#include <stdio.h>
int main()
{
    char ch1 = 'x';
    char ch2 = 66;
    char ch3 = 'A' + 2;;

    // Ausgabe als Zeichen
    printf("%c %c %c\n", ch1, ch2, ch3);

    // Ausgabe als Zahlen
    printf("%d %d %d\n", ch1, ch2, ch3);

    // Sonderzeichen
    printf("\"\\t\\'\\t\\\\t\\?\\n");

    // Klingelzeichen
    printf("\\a");

    // Zeichenkette über mehrere Zeilen
    printf("Hello "
           "World" "\\n");

    return 0;
}
```

Und was
mach' ich
bis zum nä.
Mal?



Visual Studio runterladen &
installieren



Beispiele aus Vorlesung
nachimplementieren



Falls Sie noch nie programmiert haben,
probieren Sie einfach mal Hour of
Code aus: <https://hourofcode.com/de>

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!