

„Systemnahe Programmierung“ (BA-INF 034) im Wintersemester 2023/2024

Dr. Matthias Frank, Dr. Matthias Wübbeling
(modulverantwortlich Prof. Dr. Peter Martini)

Institut für Informatik 4
Universität Bonn
e-mail: {matthew, matthias.wuebbeling} @cs.uni-bonn.de

Herzlich Willkommen!!!

**Vorlesung wöchentlich
Dienstags von 14.15 Uhr bis 15.45 Uhr
in HS-2 Hörsaalzentrum**

Besonderheiten im WS 2023/24 (... nach „Corona“)

Bereits das vergangene WS 2022/23 war von der Leitung der Uni Bonn als „Präsenzsemester“ angekündigt.

- Auswirkungen auf die **Vorlesungen**

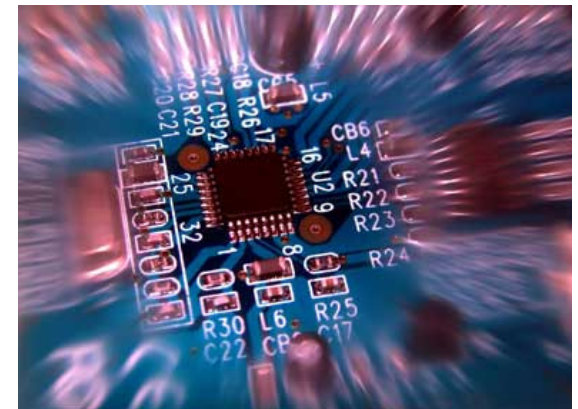
- Wöchentliche Vorlesung Dienstags 14.15 – 15.45h in HS-2 nur noch in Präsenz
- KEIN Live-Stream
- Zugriff auf die (alten) Aufzeichnungen vom WS 2021/2022 ist im (begründeten) Bedarfsfall möglich

- Auswirkungen auf die **Übungen**
 - Wöchentliche Programmierübungen in Kleinst-Gruppen (3 Studierende)
 - Schwerpunkt auf Präsenz: 12 von 13 Gruppenterminen in Präsenz
 - 1 von 13 Gruppenterminen (dauerhaft) als Online-Übung in BBB-Konferenz
 - Vorführung der programmierten Lösungen im CIP-Pool (Präsenz-Gruppen) oder am eigenen Rechner in Bildschirmfreigabe (BBB-Gruppen)
 - Tutor organisiert Treffen in Präsenz bzw. BBB
- Auswirkungen auf die **Prüfungen**
 - Geplant ist einheitlich eine Klausur in Präsenz im Hörsaal, übliche Termine Februar/März 2024
 - Zu Details (Termine, Modalitäten, ...) informieren wir, sobald bekannt

(alle Planungen vorbehaltlich Änderungen wg. Corona-Vorgaben)

Systemnahe Programmierung

- Low-level (systemnah)
 - Programmierer hat **direkten Zugriff auf Hardware und Betriebssystemschnittstellen**
 - Manuelles Speichermanagement (direkter Zugriff auf Bits und Bytes)
 - Hohe Performance und hohe Anfälligkeit
 - Nur eingeschränkt plattformunabhängig
 - Typischerweise C (C++)
- Unsere Vorlesung behandelt Konzepte wie
 - Speichermanagement
 - Parallelität und Nebenläufigkeit (Concurrency)
 - Synchronisierung
 - Interprozesskommunikation (IPC)
 - Netzwerkprogrammierung
 - C als Programmiersprache
(Rust wird historisch noch mit erwähnt)
(... und auch noch Go, gleich ganz kurz ;-)



Warum noch C?



AIPro (seit WS2019/2020) ggf. andere Sprache, aber **Grundlagen imperative Programmierung** von dort bekannt.

AIPro in diesem Jahrgang (WS 2022/2023) hatte **Einführung in C (incl. Pointer)**, Studierende Cyber Security hatten auch **C-Intro in Grundlagen der IT-Sicherheit**.

IEEE SPECTRUM Top 10 der Programmiersprachen - 2019

Quelle:

<https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2019>





Engineering Topics ▾

Special Reports ▾

Blogs ▾

Multimedia ▾

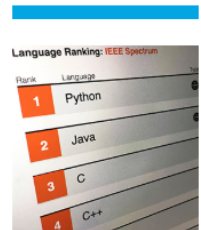
The Magazine ▾

Rank	Language	Type	Score
1	Python	  	100.0
2	Java	  	96.3
3	C	  	94.4
4	C++	  	87.5
5	R		81.5
6	JavaScript		79.4
7	C#	   	74.5
8	Matlab		70.6





von 2019

Interactive: The Top Programming Languages

This app ranks the popularity of dozens of programming languages. You can filter them by excluding sectors that aren't relevant to you, such as "Web" or "Embedded." (The sectors that languages are assigned to are based on typical use patterns we've seen in the wild, rather than atypical or proof-of-concept projects.) Rankings are created by weighting and combining 11 metrics from 8 sources—CareerBuilder, Google, GitHub, Hacker News, the IEEE, Reddit, Stack Overflow, and Twitter. ([Read more about our method and sources](#)).



Click here to read about the trends shaping this year's Top

9	Swift	 	69.1
10	Go	 	68.0
11	Arduino		67.2
12	HTML,CSS		66.8
13	PHP		65.1
14	Assembly		63.7
15	SQL		63.4
16	Dart	 	57.4
17	Rust	  	55.5

IEEE SPECTRUM Top 10 der Programmiersprachen - 2020

Quelle:

<https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2020>



Engineering Topics ▾

Special Reports ▾

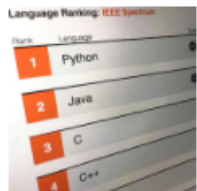
Blogs ▾

Multimedia ▾

The Magazine ▾

Interactive: The Top Programming Languages

This app ranks the popularity of dozens of programming languages. You can filter them by excluding sectors that aren't relevant to you, such as "Web" or "Embedded." (The sectors that languages are assigned to are based on typical use patterns we've seen in the wild, rather than atypical or proof-of-concept projects.) Rankings are created by weighting and combining 11 metrics from eight



Click here to read about the trends shaping this year's Top Programming Languages

Rank	Language	Type	Score
1	Python ▾	🌐 🖥️ ⚙️	100.0
2	Java ▾	🌐 📱 🖥️	95.3
3	C ▾	📱 🖥️ ⚙️	94.6
4	C++ ▾	📱 🖥️ ⚙️	87.0
5	JavaScript ▾	🌐	79.5
6	R ▾	🖥️	78.6
7	Arduino ▾	⚙️	73.2
8	Go ▾	🌐 🖥️	73.1

C unverändert Platz 3
Go von 10 auf 8
Rust von 17 auf 20

9	Swift ▾	📱 🖥️	70.5
10	Matlab ▾	🖥️	68.4
11	Ruby ▾	🌐 🖥️	66.8
12	Dart ▾	🌐 📱	65.6
13	SQL ▾	🖥️	64.6
14	PHP ▾	🌐	63.8
15	Assembly ▾	⚙️	63.7
16	Scala ▾	🌐 📱 🖥️	63.5
17	HTML ▾	🌐	61.4
18	Kotlin ▾	🌐 📱	57.8
19	Julia ▾	🖥️	56.0
20	Rust ▾	🌐 🖥️ ⚙️	55.6

, Hacker News, Twitter. (Read special thanks to API, which

IEEE Spectrum
hose more

IEEE SPECTRUM Top 10 der Programmiersprachen - 2021

Quelle:





















<https://spectrum.ieee.org/top-programming-languages/>

<https://spectrum.ieee.org/top-programming-languages-2021>

INTERACTIVE

Top Programming Languages 2021

This app ranks the popularity of dozens of programming languages. You can filter them by excluding sectors that aren't relevant to you, such as "Web" or "Embedded." (The sectors that languages are assigned to are based on typical use patterns we've seen in the wild, rather than atypical or proof-of-concept projects.)

Rank	Language	Type	Score
1	Python	  	100.0
2	Java	  	95.4
3	C	  	94.7
4	C++	  	92.4
5	JavaScript		88.1
6	C#	   	82.4
7	R		81.7
8	Go	 	77.7

ited by weighting and combining 11 metrics from eight sources: [CareerBuilder](#), [GitHub](#), [LeetCode](#), [news](#), the [IEEE](#), [Reddit](#), [Stack Overflow](#), and [Twitter](#). ([Read more about our method](#) and thanks to [CareerBuilder](#) for letting us use its data.

weights produces our *IEEE Spectrum* ranking, but there are preset weights for those more it's trending or most looked for by employers. Don't like the presets? Make your own ranking veights yourself using the "Create Custom Ranking" option.


Choose a Ranking

IEEE Spectrum


Trending

















Language Types

Web



Enterprise



9	HTML		75.4
10	Swift	 	70.4
11	Arduino		68.4
12	Matlab		68.3
13	PHP		68.0
14	Dart	 	67.7
15	SQL		65.0
16	Ruby	 	63.6
17	Rust	  	63.1
18	Assembly		62.8
19	Kotlin	 	58.5
20	Julia		58.3

Read also

Learn more about the trends shaping this year's Top Programming Languages

ANALYSIS

COMPUTING


Top Programming Languages 2021

ANALYSIS

COMPUTING

IEEE Top Programming Languages: Design, Methods, and Data Sources

TOP PROGRAMMING LANGUAGES IS SPONSORED BY



The NYU Tandon School of Engineering is one of the most respected schools of engineering, applied sciences, and management in the U.S. Visit [NYU Tandon](#) for more information on undergraduate, graduate, and online degree programs.

C unverändert Platz 3

Go unverändert auf 8

Rust wieder zurück von 20 auf 17

IEEE SPECTRUM Top 10 der Programmiersprachen - 2022

Quelle:

<https://spectrum.ieee.org/top-programming-languages/>

<https://spectrum.ieee.org/top-programming-languages-2022>

Top Programming Languages 2022 > Python's still No. 1, but employers love to see SQL skills

BY STEPHEN CASS | 23 AUG 2022 | 4 MIN READ |

Related Stories

ARTICLE HISTORY OF TECHNOLOGY

The Electric Purple Snake-Oil Machine

HANDS ON DIY

Upcycling a 40-year-old Tandy Model 100 Portable Computer

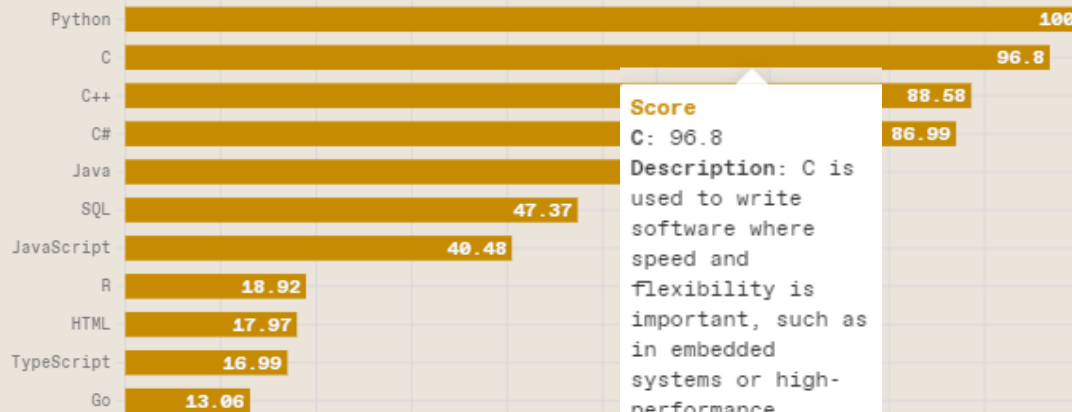
NEWS COMPUTING

Coding Made AI—Now, How Will AI Unmake Coding?

Top Programming Languages 2022

Click a button to see a differently weighted ranking

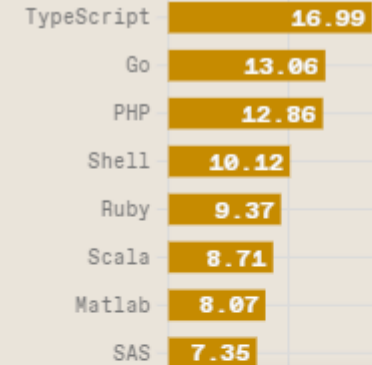
Spectrum Jobs Trending



Score

C: 96.8

Description: C is used to write software where speed and flexibility is important, such as in embedded systems or high-performance computing.



Score

Rust: 5.01

Description: Rust is designed to make concurrent systems easier to program reliably.

C, C++ und C# vorbei an Java auf Plätzen 2,3,4

Go runter auf Platz 11

Rust wieder runter von 17 auf 20 (wie 2020)

IEEE SPECTRUM Top 10 der Programmiersprachen - 2023

Quelle:

<https://spectrum.ieee.org/top-programming-languages/>

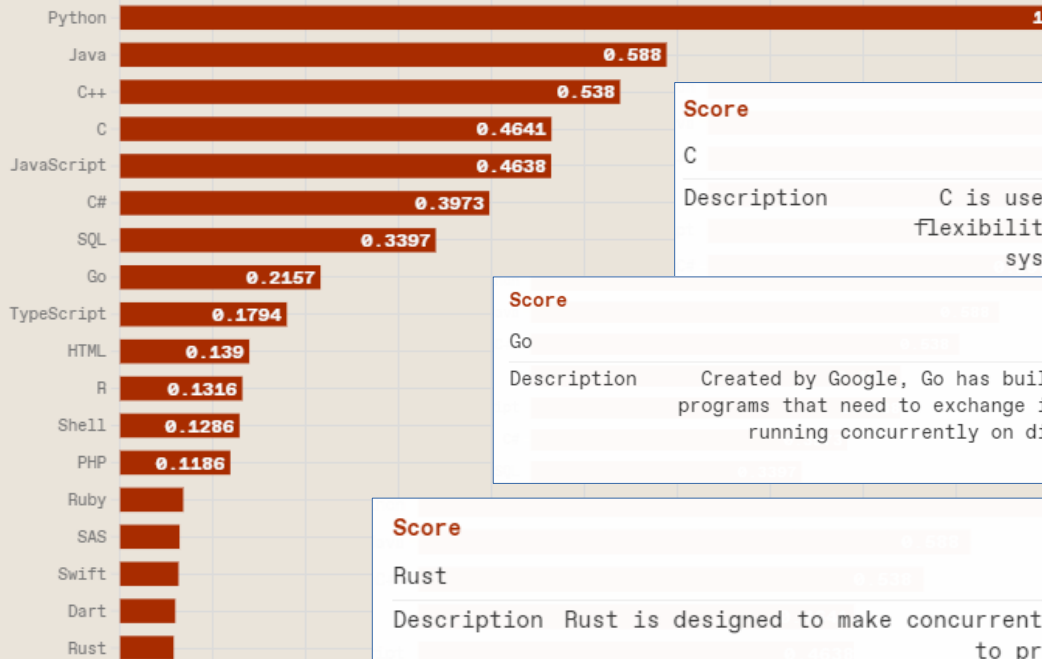
<https://spectrum.ieee.org/top-programming-languages-2023>



Top Programming Languages 2023

Click a button to see a differently weighted ranking

Spectrum Jobs Trending



Score

C

0.4641

Description

C is used to write software where speed and flexibility is important, such as in embedded systems or high-performance computing.

Score

Go

0.2157

Description

Created by Google, Go has built-in support for programs that need to exchange information while running concurrently on different cores or computers.

Score

Rust

0.0576

Description

Rust is designed to make concurrent systems easier to program reliably.

C++, C, und C# hinter Python und Java auf Plätzen 3,4,6
Go von Platz 11 auf Platz 8
Rust von Platz 20 auf Platz 18

Historie der Sys-Prog (BA-INF 034)

- **Wintersemester 2008/2009 erste Durchführung** dieser Vorlesung im Bachelor-Studiengang
 - Prof. Marrón und Dr. Frank
 - Prof. Marrón hat zum WS09/10 zur Uni Duisburg gewechselt.
- **Wintersemester 2009/2010**
 - Dr. Frank + AG Martini
- **Wintersemester 2010/2011**
 - Professurvertretung durch Dr. Meier
- **Wintersemester 2011/2012**
 - Prof. Björn Scheuermann und Mitarbeiter (mittlerweile HU Berlin)
- **Wintersemester 2012/2013 + 2013/2014**
 - Prof. Dr. Michael Meier, Dr. Matthias Frank und Mitarbeiter
- **Wintersemester 2014/2015**
 - Prof. Dr. Matthew Smith, Dr. Matthias Frank und Sergej Dechand
 - Bewährtes behalten, Interessantes/Neues mit hinzu
- **WS 2015/2016:** „Leichte Evolution“ (z.B. **Go** hinzu)
- **WS 2016/2017:** wie Vorjahr
- **WS 2017/2018:** Personaländerung: + Stephan Plöger
- **WS 2018/2019:** **Go durch Rust ersetzt**
- **WS 2019/2020:** Assembler incl. 64Bit
- **WS 2020/2021:** (Corona) inhaltlich wie Vorjahr; + Matthias Wübbeling
- **WS 2021/2022:** wie Vorjahr; wieder Präsenz + hybrid
- **WS 2022/2023:** Schwerpunkt Präsenz (+ Vorl. in BBB live)
- **WS 2023/2024:** Vorlesung Präsenz, 1 Übung online

Zur Person: Dr. Matthias Frank

1988 – 1994 **Studium** der Informatik an der
Uni-GH Paderborn

1994 – 1996 **Wissenschaftlicher Mitarbeiter** an der
Uni-GH Paderborn
AG Rechnernetze, Prof. Dr. Peter Martini

1996 – 1999 **Wissenschaftlicher Mitarbeiter** an der Uni Bonn
Institut für Informatik IV, Prof. Dr. Peter Martini

1999 **Promotion** zum Dr. rer. nat.

Seit 1999 **Akademischer Rat/Oberrat** auf einer Dauerstelle
am Inst. f. Inform. IV, Uni Bonn

WS 2002/2003 Lehrstuhlvertretung im Fachgebiet „Rechnernetze“
+ SS 2003 an der Uni Paderborn

Seit Oktober 2003 wieder hauptamtlich an der Uni Bonn



Vermutlich bekannt
aus **Systemnahe
Informatik SS23**
(aka Matthew)

Lehrtätigkeit: Dr. Matthias Frank

Vorlesungen (und teilweise Übungen) im Hauptstudium + Master ab WS2008/2009

- **Kommunikationsprotokolle** und Mobile Vernetzung I u. II
- **Mobilkommunikation**
- **Rechnernetze**/Data Communication & Internet Technology

Im Rahmen der **Lehrstuhlvertretung** thematisch ähnliche Vorlesungen + Seminar

Vermutlich bekannt
aus **Systemnahe
Informatik SS23**
(aka Matthew)

Für das **Grundstudium Informatik + Bachelor ab WS2008/2009**

- mehrfach **Proseminar** „Internetprotokolle/Mobilfunkprotokolle/Vert. Systeme“
- mehrfach **Programmierpraktikum** „Rechnernetze / Kommunikationssysteme“
- sowie „Steuern und Schalten mit dem Computer“
- SS 2004 +SS 2005: **Vorl. Grundlagen der Programmierung II** (nach Diplom PO2003)
- ab WS 2008/2009 **Vorl. Systemnahe Programmierung** (BA-INF 034)
- ab SS 2009 **Projektgruppen im Themenbereich Sys-Prog/Komm.systeme** (BA-INF 051)
- regelmäßig gemeinsam mit Prof. Martini **Vorl. Systemnahe Informatik** (BA-INF 032)
- regelmäßig gemeinsam mit Prof. Martini **Vorl. KIVS – Kommunikation
in verteilten Systemen** (Wahlvorlesung BA-INF 101)

Zur Person: Dr. Matthias Wübbeling

- 2011 Abschluss Diplom-(Kern-)Informatik an der Technischen Universität Dortmund
- 2011 **Wissenschaftlicher Mitarbeiter** an der TU Dortmund
FG Maschinenelemente, Prof. Dr. habil. Künne
- Seit 2012 **Wissenschaftlicher Mitarbeiter** am Fraunhofer FKIE
- 2012 - 2019 **Wissenschaftlicher Mitarbeiter** und **Doktorand** an der Universität Bonn, AG IT-Sicherheit, Prof. Dr. Meier
- Seit 2015 **Freiberuflich Autor in Fachzeitschriften und Berater**
- 2019 **Promotion** zum Dr. rer. nat.
- Seit 2020 **Akademischer Rat** auf einer Dauerstelle am Institut für Informatik IV der Universität Bonn, AG IT-Sicherheit, Prof. Dr. Meier
- Seit 2020 **Geschäftsführender Gesellschafter** der Identeco GmbH & Co. KG
(Ausgegründet aus dem Forschungsprojekt „Effektive Information nach Digitalem Identitätsdiebstahl - EIDI)

Vorlesungen / Übungen im Bachelor und Master ab SS 2012

- Systemnahe Informatik Übung (SS 2012, 2013, 2014, 2015)
mit Dr. Frank & Prof. Martini
- Systemnahe Programmierung Übung (WS 2012/13, 2013/14)
mit Dr. Frank & Prof. Meier
- IT-Security Vorlesung (SS 2015, 2016, 2017, 2018, 2019, 2020)
mit Prof. Meier

Eigenständige Lehre ab SS 2020 **Informatik & Cyber Security Bachelor**

- **Vorlesung Netzwerksicherheit** (Wahlpflicht)
- **Vorlesung Systemnahe Programmierung** (Wahlpflicht / Pflicht)

- Forschungsinteressen
 - **Unterschiedliche Aspekte der IT-Sicherheit**
 - Bedrohungsanalyse / Threat Intelligence
 - Datenschutz (Pseudonymisierung / Anonymisierung)
 - IT-Security-Awareness
 - Reaktive Sicherheit
 - Netzwerksicherheit (IDS/IPS, Anomalieerkennung)
 - Sicherheit im Internetrouting
 - (Präventiver) Schutz vor Identitätsdiebstahl
- Lehrtätigkeiten
 - Vorlesung „**AIPro**“
 - Vorlesung „**IT-Sicherheit**“
 - Vorlesung „**Systemnahe Programmierung**“
 - Vorlesung „**Reaktive Sicherheit**“
 - Vorlesung „**Netzwerksicherheit**“
 - Master-Vorlesung „**IT-Security**“
 - Master-Labs „**IT-Security**“
 - Seminar: **Selected Topics in IT-Security**

Studienberatung Informatik

Beratung und Beantwortung bei Studienfachspezifischen Fragen

<https://www.informatik.uni-bonn.de/de/fuer-studierende/studienberatung>

Sie sind hier: [Startseite](#) → [Für Studierende](#) → Studienberatung

Studienberatung

erstellt von massimo — zuletzt verändert: 26.04.2021 11:39

Informatik

Unser Studienberater Dr. Matthias Frank berät Studieninteressierte und alle Studierenden an unserem Institut zum Fach Informatik. Um einen Termin zu vereinbaren, kontaktieren Sie ihn bitte per Telefon oder E-Mail.

Dr. Matthias Frank



Tel: 0228-73-4550

E-Mail: studienberatung@cs.uni-bonn.de

Raum 1.004

Friedrich-Hirzebruch-Allee 8

Postanschrift: Friedrich-Hirzebruch-Allee 5

53115 Bonn

www: <https://net.cs.uni-bonn.de/de/admin/studienberatung/studienberatung-informatik>

Cyber Security

Unser Studienberater Dr. Matthias Wübbeling berät Studieninteressierte und Studierende an unserem Institut zum Fach Cyber Security. Um einen Termin zu vereinbaren, kontaktieren Sie ihn bitte per Telefon oder E-Mail.

Dr. Matthias Wübbeling

Tel: 0228-73-54250

E-Mail: studienberatung@cs.uni-bonn.de

Raum: 1.016

Friedrich-Hirzebruch-Allee 8

Postanschrift: Friedrich-Hirzebruch-Allee 5

53115 Bonn

Bei Fragen:

In Präsenz auch ideal einfach
nach der Vorlesung ...

(sonst E-Mail, ggf. Vereinbarung
eines Termins)

Und wer sind Sie / wer seid Ihr ... ?

- ... insbesondere wie viele? ;-)

Nicht vergessen:

Umfrage/Abzählen im Hörsaal

Gesamtanzahl:	
BA Informatik	
BA Cyber Security	
Lehramt Informatik	
FFF-ler	
anderes Fach	
Nicht eingeschrieben	

Organisatorisches + Allgemeines

Wer kann/darf/muss an der Veranstaltung Systemnahe Programmierung teilnehmen ???

Die Vorlesung „Systemnahe Programmierung“ (BA-INF 034) gibt es seit der Bachelor-PO von 2007, vorgesehen im 3. Semester (erster Betrieb ab WS 2008/2009), zunächst Pflicht-Modul

Mittlerweile
Wahlpflicht!

=> **prüfungsrelevant nur für Informatik Bachelor-Studierende**
(ggf. Ausnahme: Förderprogramm FFF)

auch: Wahlpflichtfach für Lehramt Informatik

auch: als Modul für Nebenfach Informatik möglich
(abhängig von PO des jeweiligen Hauptfaches)

NEU: Pflichtmodul 3. Sem. für Bachelor Cyber Security

Pflichtmodul
nur für
Cyber Security

Alle weiteren „freiwilligen“ HörerInnen sind herzlich willkommen!

(eine Teilnahme an den Übungen mit Bewertung der Abgaben/Vorführungen ist jedoch nur bei einer entsprechend niedrigen Gesamtteilnehmerzahl möglich)

Zulassung zur Teilnahme am Modul „Systemn. Prog.“

Teilnahmevoraussetzungen:

Lt. Modulhandbuch nur eine Empfehlung:

Systemnahe Informatik (BA-INF 023)

Die Inhalte der SysInf werden als bekannt vorausgesetzt, hilfreich z.B.

- Alpha-Notation für Assembler
- Synchronisation/Threads

Wichtig:

Wir setzen die Bereitschaft voraus, sich selbständig im nötigen Umfang in C einzuarbeiten.

Teilnehmer/innen der VL IT-Sicherheit hatten bereits eine C-Einführung, ebenfalls AIPro Jahrgang WS 2022/2023.

Bachelor Informatik — Universität Bonn						20
Modul BA-INF 034		Systemnahe Programmierung				
Workload 180 h		Umfang 6 LP	Dauer 1 Semester	Turnus jährlich		
Modulverantwortlicher		Prof. Dr. Peter Martini				
Dozenten		Dr. Matthias Frank, Prof. Dr. Matthew Smith				
Zuordnung		Studiengang B. Sc. Informatik 2019	Modus Wahlpflicht	Studiensemester 3.		
Lernziele: fachliche Kompetenzen		Die Studierenden sollen in der Lage sein, Techniken der system- und maschinennahen Programmierung (d.h. verteilte, parallele, ereignisorientierte sowie prozessnahe Programmierung) angemessen und im Detail realisieren zu können.				
Lernziele: Schlüsselkompetenzen		Ein Schwerpunkt in den unterstützenden Übungen liegt in der praktischen Umsetzung in Kleingruppen (Teamfähigkeit) sowie der Diskussion und dem Vorstellen eigener Lösungen				
Inhalte		Netzwerk-/Socket-Programmierung (in C/C++), Input-Output-Multiplexing, Serverstrukturen, verteilte Programmierung (Remote Method Invocation), Shared-Memory-/Thread-Programmiersmodelle, Specification and Description Language (ereignisorientierte Programmierung), Fortgeschrittene Konzepte von Nebenläufigkeit, u.a. Channels, Coroutinen, Share-Memory-by-Communicating, Dynamic Memory Allocation und Memory Pooling; Ausdrucksprogrammierung in Assembler				
Teilnahmevoraussetzungen		Empfohlen: BA-INF 023 – Systemnahe Informatik				
		Lehrform	Gruppengröße	SWS	Workload[h]	LP
Veranstaltungen		Vorlesung Übungen		2 2	90 P / 45 S 30 P / 75 S	2,5 3,5
		P = Präsenzstudium, S = Selbststudium				
Prüfungsleistungen		Schriftliche Prüfung				(benotet)
Studienleistungen		Erfolgreiche Übungsteilnahme				(unbenotet)
Medieneinsatz						
Literatur		<ul style="list-style-type: none">• C. A. R. Hoare: Communicating Sequential Processes, Prentice Hall International, Electronic version 2004 edited by Jim Davies, http://www.usingsp.com/cspbook.pdf• W. Richard Stevens et al.: UNIX Network Programming – The Sockets Networking API, Prentice Hall International, 3rd Edition, 2003• Andrew S. Tanenbaum, Maarten van Steen: Distributed Systems: Principles and Paradigms, Prentice Hall International 2006• Markus Zahn: UNIX-Netzwerkprogrammierung mit Threads, Sockets und SSL, Springer 2006 Weitere Literaturhinweise werden rechtzeitig vor Vorlesungsbeginn bekannt gegeben.				

Prüfungsmöglichkeit: Klausur

Die Vorlesung „Systemnahe Programmierung“ wird mit einer **schriftlichen Modulprüfung (Klausur)** am/nach Ende der Vorlesungszeit geprüft.

Voraussetzungen:

- Erfolgreiche Teilnahme an den **Übungen zu Systemnahe Programmierung** (WS 23/24) – oder Übungserfolg/Zulassung aus vorherigen Semestern

Termine:

die erste Prüfungsmöglichkeit des Moduls: **tba.**
(irgendwann im Februar 2024)

Wiederholungsmöglichkeit der Modulprüfung: **tba.**
(ca. irgendwann im März 2024)

Die Klausuren sind wieder in Präsenz.

Stundenplan der Vorlesung Systemnahe Programmierung

Reguläre Vorlesungstermine

Erster Termin: heute Dienstag 10.10.2023

Dienstags 14.15 - 15.45 Uhr Im Hörsaal in Präsenz

Letzter Termin: Dienstag 31.01.2023

Insgesamt 15 Wochen Vorlesungen (15x) und Übungen (13x + ggf. 1):

- in der ersten Vorlesung 10.10. Organisatorisches + Einleitung
- in der ersten und zweiten Woche noch keine Übungen/Vorfürhrungen
- Erstes Übungsblatt schon am Mi 11.10.2023
- in der dritten Woche Mo 23.10.2022 erste Übungen (Kennenlernen + Blatt 1)
- ab der vierten Woche Mo 30.10.2022 regelmäßige Übungen/Vorfürhrungen
- Mittwoch Feiertag **1.11.2023** keine Übungen
- Sa 23.12.2023 bis Sa 06.01.2024 kein Vorlesungs-/Übungsbetrieb
- d.h. Woche 18.12. - Fr 22.12.2023 noch regulär Vorlesung + Übungen
- Übungsbetrieb läuft normal in den „vollen“ Wochen vor Weihn. bis Fr 22.12.2023
- und wieder ab Mo 08.01.2024 (Details dazu dann auch auf den Übungszetteln)

Infos & Material zu Systemnaher Programmierung

Einfach im Web nachschauen unter

<http://net.cs.uni-bonn.de/teaching/wt-202324/>

dort bei „Bachelor“ – „Modul Systemnahe Programmierung...“

Dort findet man

- Folienkopien / Präsentationen
- Übungszettel

Aufgepasst: Von außerhalb der Uni wird eine Nutzerkennung verlangt!

Login: *****
Password: *****

Angekündigt in der
Vorlesung am 10.10.2023

(später bitte Komiliton/innen
o. Tutoren/Doz. fragen)

Aktuelle Infos rund um Vorlesung und Übung gibt es im **Mailverteiler „vl-sys-prog“**:

➡ Dort wird jeder eingetragen, der sich für die Übung anmeldet!

➡ Details zur Übungsanmeldung folgen später.

Ankündigungen zum Übungsbetrieb
auf separaten Folien (jetzt!)

=> diese sind (ebenfalls) auf den Webseiten zu finden
unter „Hinweise zum Übungsbetrieb“

Inhaltliches zur Systemnahen Programmierung

Bachelor Informatik — Universität Bonn

20

Modul BA-INF 034	Systemnahe Programmierung				
Workload 180 h	Umfang 6 LP	Dauer 1 Semester	Turnus jährlich		
Modulverantwortlicher	Prof. Dr. Peter Martini				
Dozenten	Dr. Matthias Frank, Prof. Dr. Matthew Smith				
Zuordnung	Studiengang B. Sc. Informatik 2019	Modus Wahlpflicht	Studiensemester 3.		
Lernziele: fachliche Kompetenzen	Die Studierenden sollen in der Lage sein, Techniken der system- und maschinennahen Programmierung (d.h. verteilte, parallele, ereignisorientierte sowie prozessornahe Programmierung) angemessen und im Detail realisieren zu können.				
Lernziele: Schlüsselkompetenzen	Ein Schwerpunkt in den unterstützenden Übungen liegt in der praktischen Umsetzung in Kleingruppen (Teamfähigkeit) sowie der Diskussion und dem Vertreten eigener Lösungen.				
Inhalte	Netzwerk-/Socket-Programmierung (in C/C++), Input-Output-Multiplexing, Serverstrukturen, verteilte Programmierung (Remote Method Invocation), Shared-Memory-/Thread-Programmiermodelle, Specification and Description Language (ereignisorientierte Programmierung), Fortgeschrittene Konzepte von Nebenläufigkeit, u.a. Channels, Coroutinen, Share-Memory-by-Communicating, Dynamic Memory Allocation und Memory Pooling; Maschinenprogrammierung in Assembler				
Teilnahme:	Empfohlen				
Voraussetzungen	BA-INF 023 – Systemnahe Informatik				
Veranstaltungen	Lehrform	Gruppengröße	SWS	Workload[h]	LP
	Vorlesung Übungen		2 2	30 P / 45 S 30 P / 75 S	2,5 3,5
	P = Präsenzstudium, S = Selbststudium				
Prüfungsleistungen	Schriftliche Prüfung (benotet)				
Studienleistungen	Erfolgreiche Übungsteilnahme (unbenotet)				
Medieneinsatz					
Literatur	<ul style="list-style-type: none">• C. A. R. Hoare: Communicating Sequential Processes, Prentice Hall International, Electronic version 2004 edited by Jim Davies, http://www.usingcsp.com/cspbook.pdf• W. Richard Stevens et al.: UNIX Network Programming – The Sockets Networking API, Prentice Hall International, 3rd Edition, 2003• Andrew S. Tanenbaum, Maarten van Steen: Distributed Systems: Principles and Paradigms, Prentice Hall International 2006• Markus Zahn: UNIX-Netzwerkprogrammierung mit Threads, Sockets und SSL, Springer 2006 Weitere Literaturhinweise werden rechtzeitig vor Vorlesungsbeginn bekannt gegeben.				

Inhalte (laut Modulhandbuch):

- Netzwerk-/Socket-Programmierung (in C/C++/Java)
- Input-Output-Multiplexing
- Serverstrukturen
- verteilte Programmierung (Remote Method Invocation)
- Shared-Memory-/Thread-Programmiermodelle
- Specification and Description Language
- (ereignisorientierte Programmierung)
- Maschinenprogrammierung in Assembler

Lernziele:

Die Studierenden sollen in der Lage sein, **Techniken der system- und maschinennahen Programmierung** (d.h. verteilte, parallele, ereignisorientierte sowie prozessornahen Programmierung) angemessen und im Detail realisieren zu können.

(Modulhandbuch Bachelor
Fassung vom SoSe 2020, für die Prüfungsordnung 2019)

„Systemnahe Programmierung“ im Überblick

Seit WS 2015/2016: „Leichte Evolution“ (Umstellung der Reihenfolge)

Seit WS 2020/2021 weitere Sprachen (erst Go, später Rust) wieder entfernt

0. Wichtige Informationen zu dieser Lehrveranstaltung

1. Maschinenprogrammierung in Assembler (**WS2019/20: mehr 64 Bit**)

- 80x86 Maschinensprache für PC-Prozessoren
- Konventionen von Funktionsaufrufen, Bsp. in C

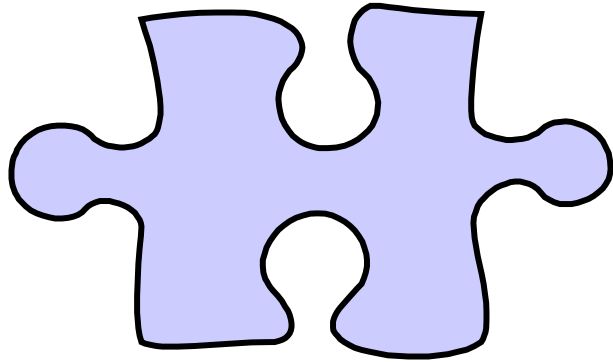
2. Fortgeschrittene Konzepte der Systemprogrammierung (in C)

- Prozesse, Threads
- IPC, Shared Memory
- Synchronisierung

3. Netzwerkprogrammierung in C

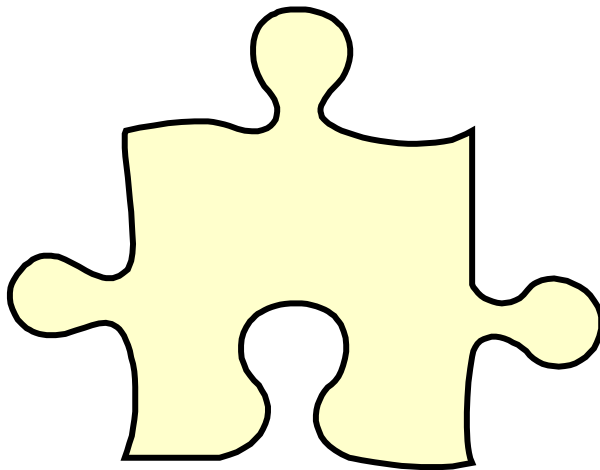
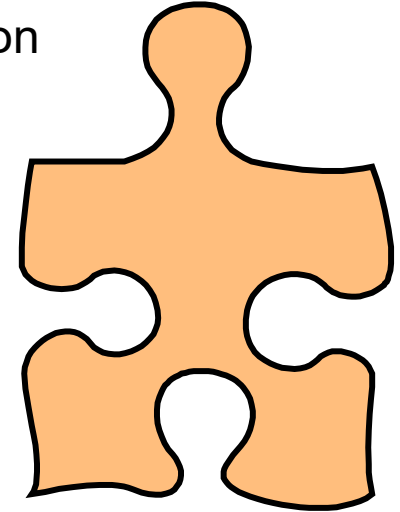
- Grundlagen zum Internet, TCP, UDP, IP
- Socketprogrammierung
- I/O-Multiplexing, Serverstrukturen

Motivation und „roter Faden“



3. Netzwerkprogrammierung in C
„ein Programm möchte über
ein Netzwerk kommunizieren“

2. Betriebssysteme/Threads
„Kommunikation innerhalb von
Prozessen bzw. zwischen
Prozessen eines Rechners“



1. Maschinenprogrammierung in Assembler
(nahe Anlehnung an Inhalte der Systemnahen
Informatik)
„wir werden hier keine Assembler-Profis, aber
Assembler-Routinen werden bei Bedarf in
Programme anderer (Hoch-) Sprachen einge-
bunden“

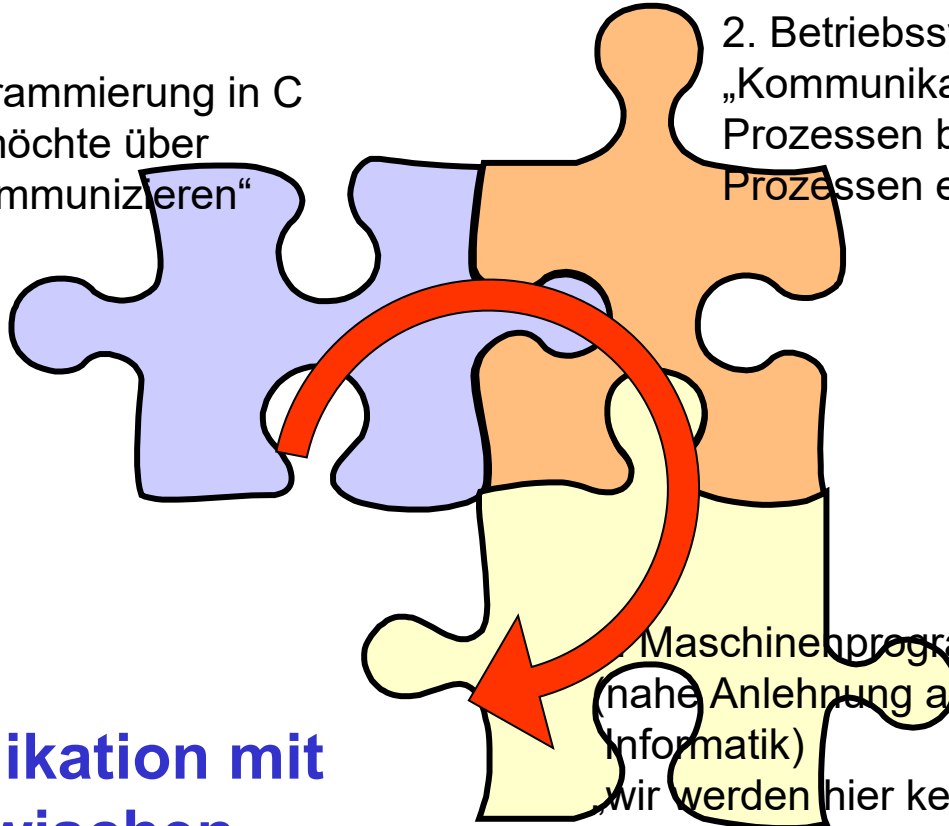
Motivation und „roter Faden“

3. Netzwerkprogrammierung in C
„ein Programm möchte über
ein Netzwerk kommunizieren“

2. Betriebssysteme/Threads
„Kommunikation innerhalb von
Prozessen bzw. zwischen
Prozessen eines Rechners“

Kommunikation mit und zwischen Prozessen/Programmen

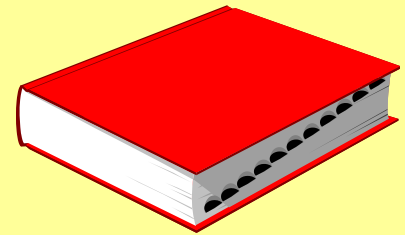
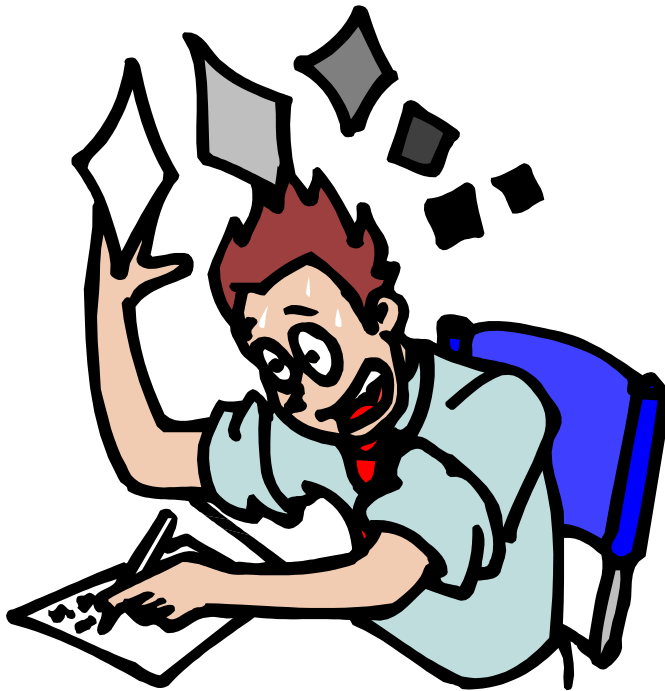
1. Maschinenprogrammierung in Assembler
(nahe Anlehnung an Inhalte der Systemnahen
Informatik)
„Wir werden hier keine Assembler-Profis, aber
Assembler-Routinen werden bei Bedarf in
Programme anderer (Hoch-) Sprachen eingebunden“



... es steht doch schon alles auf den Folien ...

Die als Hilfsblätter zur Vorlesung im Internet verfügbaren **Folien können eigene Notizen nicht ersetzen !!**

Machen Sie unbedingt **erläuternde Anmerkungen auf den Folienausdrucken**. Spätestens bei der Vorbereitung zu einer Prüfung werden Sie den **Wert eigener, ergänzender Aufzeichnungen** zu schätzen wissen*.



Literatur sowie Hinweise auf **Online-Informationen** werden in den einzelnen Kapiteln bekannt gegeben.

*Es ist wichtig, die Erstellung sinnvoller Mitschriften zu erlernen und zu üben, da diese Fähigkeit auch nach Abschluss des Studiums bei Weiterbildungsveranstaltungen unverzichtbar ist.