Organisation und Inhalt

Manfred Hauswirth | Open Distributed Systems | Einführung in die Programmierung, WS 23/24

Wer sind wir?

- Fachgebiet „Open Distributed Systems“ (ODS)
  - Leitung: Prof. Manfred Hauswirth
- Veranstalter der Vorlesung ”Einführung in die Programmierung”
- Sie begegnen uns vor allem als:
  - Prof. Manfred Hauswirth (Vorlesungen)
  - Wissenschaftliche Mitarbeiter (ISIS)
  - TutorInnen (Lehraufgaben)
- Wenn Sie Fragen oder Probleme haben:
  - Zur Immatrikulation? → Campus Center
  - Zum TUB-Account? → ZECM
  - Zum Kurs? → ISIS

Prof. Manfred Hauswirth

Technische Universität Berlin

Fachgebietsleiter „Open Distributed Systems“ – https://www.tu-berlin.de/ods

Institutsleiter Fraunhofer FOKUS – https://www.fokus.fraunhofer.de

- Skalierbare verteilte Informationssysteme
- Linked Data-Stromdatenverarbeitung
- Quantencomputing
- Semantische Sensor-Netzwerke
- Semantic Web
- Peer-to-Peer-Systeme

Open Distributed Sytems ODS

Melanie Lahrkamp

Technische Universität Berlin

Fachgebietsassistenz

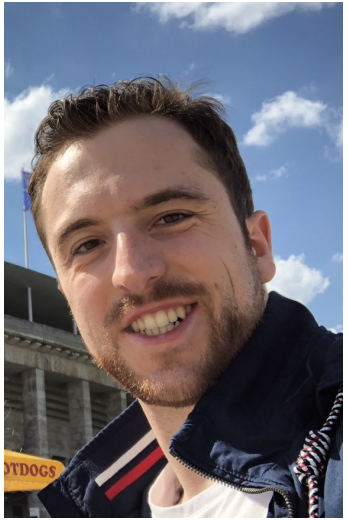
Kontakt: sekretariat@ods.tu-berlin.de

Open Distributed Sytems
ODS

Damien Foucard

Technische Universität Berlin

Wiss. Mitarbeiter / Dissertant



- Hauptthema: Heavy Hitter Monitoring
  - „Viele Daten, wenig Zeit. Was ist wichtig?“
- Subthemen:
  - Trend Analysis on Texts
  - Network Monitoring
  - Recommendations on Graphs
- Stärken:
  - Statistik
  - Algorithmik

Open Distributed Sytems
ODS

Aljoscha Meyer

Technische Universität Berlin

Wiss. Mitarbeiter / Dissertant

- Peer-to-peer Systeme
- Datensynchronisation
- Kommuniationsprotokolle
- theoretische Informatik

Open Distributed Sytems
ODS

Uwe A. Kuehn

Technische Universität Berlin

Wiss. Mitarbeiter / Dissertant

- Verteilte und hybride DBMS
- Datenreplikation, -partitionierung und -synchronisierung
- Applikationen auf begrenzten Ressourcen z.B. Raspberry Pi
- Blockchain, Smart Contracts
- Distributed Quantum Computing

Open Distributed Sytems
ODS

Wo sind wir?

HFT, 4. Stock, Raum 411

Informationen und Kontakte

- Infos über ISIS
- Forum (ISIS)
- E-Mail: intoprog@ods.tu-berlin.de
- Kontakt **nur über die obige E-Mail-Adresse, nicht** individuell (damit Sie **sicher** eine Antwort bekommen)

Studiengänge

- Informatik B.Sc.
- Technische Informatik B.Sc.
- Medieninformatik B.Sc.
- Medientechnik B.Sc.
- ...

Lernziele

Lernziele

- Kenntnisse
  - elementarer Datenstrukturen
  - elementarer Such- und Sortierverfahren

- Fähigkeiten
  - Probleme und Strukturen (wieder) zu erkennen
  - für ein gegebenes Anwendungsproblem die geeignete Datenstruktur zu wählen

Lernziele

- Verständnis des Paradigmas der imperativen Programmierung

- Fähigkeiten
  - einfache Programme schreiben
  - lesbare und verständliche Programme schreiben
  - den Aufwand (Komplexität) eines Algorithmus bzw. eines Programms abschätzen

Lernziele

- Einführung in eine Programmiersprache
  - Elementare Datentypen und Operatoren
  - Kontrollstrukturen: Verzweigungen, Schleifen
  - Funktionen
  - Dynamische Datenstrukturen

- Datenstrukturen
  - Listen
  - Queue (Warteschlange), Stack (Stapel) und Heap (Haufen)
  - Bäume

Lernziele

- Elementare Algorithmen
  - Suchen
  - Sortieren
- Algorithmen
  - Aufwandsabschätzung
  - Korrektheit

Lernziele

- 2 Schwerpunkte entsprechend der „Werkzeugklassen“
  - Erlernen einer Programmiersprache (hier die Sprache C)
  - Umgang mit Datenstrukturen und algorithmischen Aspekten

- Entsprechend 2 Vorlesungsteile
  - Programmierkurs (täglich in den ersten 2 Vorlesungswochen)
  - Einführung in die Programmierung (IntroProg) – wöchentliche Vorlesung

- betreutes Arbeiten

Lernziele

- Beispiel-Programmiersprache C
  - weit verbreitet, etabliert – Z.B. sind in C programmiert

    Windows, Linux, MacOS, Android, iOS, Oracle, MySQL, MS SQL Server, Web Server, Embedded Systems, Internet of Things, etc., etc., etc.
  - auf allen Plattformen verfügbar
  - Grundlage für viele weitere Vorlesungen, u.a. Rechnerorganisation
- Hier:
  - Programmierung „im Kleinen“
  - Algorithmisches „Handwerkszeug“
- Programmbeispiele auf Deutsch und/oder Englisch

Ablauf

Ablauf im Detail

Diese Veranstaltung besteht aus 2 Teilen:

1. **Programmierkurs**
   - Vorstellung der Konzepte
   - **Blockveranstaltung (täglich), 16.10. – 27.10.2023, 12:15 – 13:45 Uhr**
   - Folgende Vorlesungen finden in diesen zwei Wochen nicht statt:
     - Rechnerorganisation
     - Informatik Propädeutikum

2. **Einführung in die Programmierung (IntroProg)**
   - Grundlegende Datenstrukturen
   - Algorithmen – am Beispiel von Listen, Bäumen, und Sortieren
   - Dauer: Rest des Semesters

Lehr- und Lernkonzept

Technische Universität Berlin

Veranstaltungen

- Vorlesung
  - Vorstellung der Konzepte
  - Beispielprogramme
- Tutorien
  - (Vor-)Besprechung der Hausaufgaben
  - Codebeispiele
- Betreute Arbeitszeiten
  - Hilfestellung beim Programmieren – inkl. Fehlersuche
- Großübung (freiwillig)
  - Q&A zu ausgewählten Themen (inverted class room)
  - Voraussetzung Vorlesungs-, und Tutoriumstoff sind durch die TN nachbereitet
  - Ggf. weitergehende Beispiele, Klausuraufgaben
  - Auswertung der Abgaben und Besprechung von Lösungsideen

Lehr- und Lernkonzept

Leistungen der Portfolioprüfung

- Hausaufgaben im Programmierkurs (Programmierung)
  - eigenständige Auseinandersetzung mit den Konzepten
  - **15% der Gesamtnote**
- Hausaufgaben während des Semesters (Programmierung und Theorie)
  - eigenständige Auseinandersetzung mit den Konzepten
  - **35% der Gesamtnote**
- Klausur am Semesterende (60min)
  - **50% der Gesamtnote**

Vorlesungstermine

Wochentag	Datum	Uhrzeit	Raum
Montag	16.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Dienstag	17.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Mittwoch	18.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Donnerstag	19.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Freitag	20.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Montag	23.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Dienstag	24.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Mittwoch	25.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Donnerstag	26.10.2023	12:15-13:45	H 0105 (Audimax) & Zoom
Freitag	27.10.2023	12:15-13:45 (bei Bedarf)	H 0105 (Audimax) & Zoom

Zoom-URL für den Programmierkurs: https://s.fhg.de/2023-Programmierkurs-Introprog

Vorlesungstermine IntroProg

- Ab **Do., 02.11.2023** regulärer Vorlesungsbetrieb
  - Vorlesung, Tutorien und Rechnerübungen (wöchentlich)
  - Einschreibung in die Tutorien in MOSES bis zum 18.10.2023 notwendig!
  - Weitere Informationen am 02.11.2023
  - Zoom-URL für die Vorlesung: https://s.fhg.de/2023-VL-Introprog

- Vorlesung: Do, jeweils 14:15 – 15:45 Uhr, H0105 (Audimax)
  - Zoom-URL für die Vorlesung: https://s.fhg.de/2023-VL-Introprog

- Diese Vorlesungen starten in der Woche vom 31.10.2023:
  - Rechnerorganisation
  - Informatik Propädeutikum

Einschreibung

- ISIS für Vorlesungsmaterial – **am besten sofort einschreiben!**
  - Wenn TUB-Account vorhanden über „Selbsteinschreibung“
  - Wenn noch kein TUB-Account vorhanden über „Gastzugang“, hier sind keine Abgaben möglich. Nach Erhalt eines TUB-Account bitte sofort einschreiben.

Anmeldefristen

- ISIS für Vorlesungsmaterial – **am besten sofort**

- Modulanmeldefrist via QISPOS oder Prüfungsamt
- Für Portfolioprüfung
  - 16.10.2023 bis 05.11.2023
  - Abmeldung bis spätestens 09.11.2023
  - Empfehlung: Wählen Sie den ersten Termin am 04.03.2024
- Bereits zur Prüfung Zugelassene dürfen die Prüfung in der (alten) schriftlichen Form ablegen:
  - 90 min Klausur am 04.03.2024
  - 16.10.2023 bis 05.11.2023
  - Abmeldung bis spätestens 04.02.2024
- **Beachten Sie die Ankündigungen in ISIS**

Prüfungsmodalitäten

- Portfolioprüfung
  - **Programmierkurs (15%)**
    + Programmieraufgaben (35%)
    + Klausur (50%)
  - Test: 04.03.2024 08:00 – 10:30 Uhr
  - Wiederholungsmöglichkeit: 27.03.2024 08:00 – 10:30 Uhr

Programmierkurs – Organisation

Programmierkurs: Tagesablauf

- Vorlesung
  - Vorstellung der Konzepte
- Tutorien
  - (Vor-)Besprechung der Hausaufgaben
  - Codebeispiele
- Betreutes Arbeiten
  - Hilfestellung beim Programmieren – inkl. Fehlersuche
- Abgaben
  - Selbstständig zu bearbeitende Programmieraufgaben
  - Einzelabgaben (keine Gruppenarbeit)
  - Die verbindliche Abgabe zur Bewertung findet im Semester statt

Programmierkurs: Tagesablauf

Zusätzlich:

- Betreutes Arbeiten, Großübungen
  - Bitte informieren Sie sich **unbedingt** über den genauen Ablauf in ISIS!
- Unterstützung per ISIS Forum
  - Hilfestellung bei (fast) allem
  - „Live“-Betreuung: während der Woche, ca. 10:00 – 20:00 Uhr
- Gegenseitige Hilfestellung im ISIS-Forum
  - Hilfestellung unter Studierenden – ohne Lösungen zu tauschen
  - Wir beantworten Fragen immer wieder, wenn wir gerade freie Kapazität haben

Programmierkurs: Tutorien

- Hilfestellung bei Problemen
  - sehr hohe Zahl an Studierenden ⇒ Bitte um Verständnis
  - Dauer: 45 Minuten

- Thema: Aktuelle Vorlesung und Aufgabenblatt
  - pro Thema gibt es mehrere Zeitwahlmöglichkeiten
  - Teilnahme an jedem Thema ist sinnvoll, aber nicht verpflichtend

- Ziel: ca. 30 Teilnehmer pro Tutorium

Programmierkurs: Tutorien-Einteilung

Verteilung der Teilnehmenden auf die Tutorien:

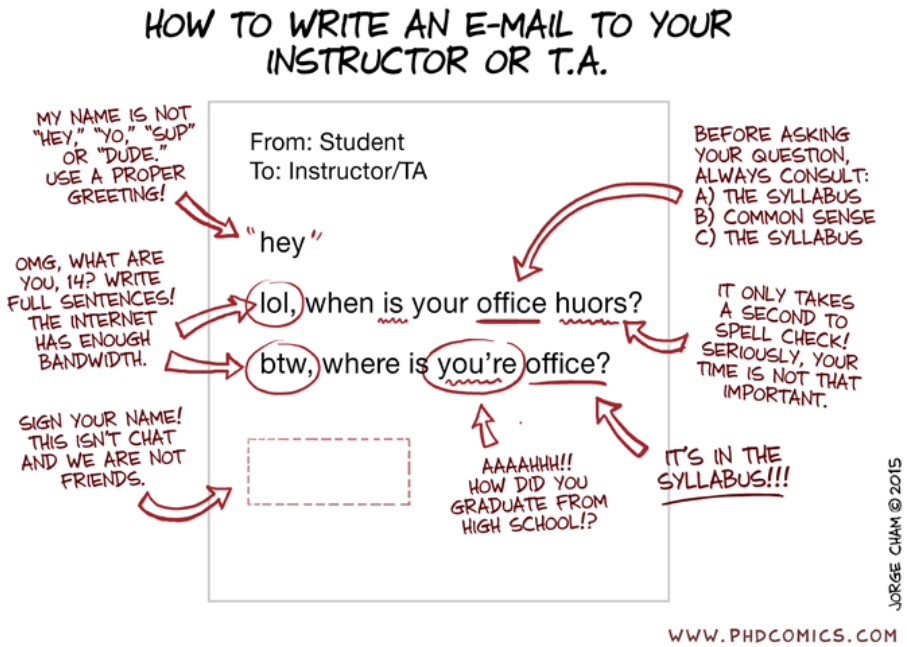
- Verfahren:
  - Ausgabe der Tutorienplätze nach Zeit
  - Es gibt begrenzte Plätze und Zeitfenster!
  - Nur belegte Tutorien finden statt!
  - Überblick über die Angebote gibt es in ISIS
- Melden Sie sich bitte über ISIS an.

Bei Problemen:

- ISIS-Forum
- **Nur bei persönlichen Problemen**: intoprog@ods.tu-berlin.de

Asking for help …

Bei Problemen: **ISIS-Forum, nur bei persönlichen Problemen**: introprog@ods.tu-berlin.de



Bewertung der Abgaben

- Fristen für die Abgaben:
  - **Unser dringender Rat:**
    So früh als möglich beginnen.
- Programmierkursblock 10 Aufgaben
  - Ausgabe nach jeder Vorlesung (ISIS)
- Semester 4 Aufgabenblöcke
  - Jede Kategorie besteht aus:
    - Programmieraufgaben
    - Theorieaufgaben (ISIS-Aktivitäten)
  - Ausgabe themenabhängig nach jeder Vorlesung (ISIS)
  - weitere Details sind auf ISIS veröffentlicht

Block	Frist	Punkte
Programmierkurs	10.11.23	15
Abgabe 1	17.11.23	6
Abgabe 2	08.12.23	8
Abgabe 3	09.01.24	12
Abgabe 4	02.02.24	8

Abgaben – Wie?

- Alle Abgaben sind beliebig oft möglich.
- Eine Aufgabe ist bestanden, wenn alle Teilaufgaben bestanden sind (keine Teilpunkte).
- Es zählt ohne Ausnahme immer die letzte Abgabe, auch „versehentliche“ oder „technisch problembehaftete“ Abgaben.
- Erfolgreiche Provisionierung des TU-Accounts erforderlich.
- ISIS-Aktivitäten (Theorie)
  - werden nach der relevanten Vorlesung geöffnet
  - schließen und sind automatisch abgegeben mit Ablauf der Abgabefrist, Vorsicht beim Wiederöffnen von bereits abgegebenen ISIS-Aktivitäten.
- Programmieraufgaben
  - Werden nach der relevanten Vorlesung zip, pdf in ISIS bereitgestellt.
  - Lösung kann in gitlab erst nach erfolgreichem „Check-In“ (s. Blatt 10) an das Testsystem übergeben werden.
  - je Aufgabe ein separater Abgabebranch (dazu mehr auf Blatt10 und in der Großübung KW44)

Einzelabgabe – wichtige Hinweise

Technische Universität Berlin

Einzelabgabe

- Jede/r Studierende erarbeitet eine eigene Lösung und gibt diese ab!
- Diskussionen von Lösungswegen, Herangehensweisen, Hilfestellung sind erlaubt und sogar erwünscht!
- Aber Weitergabe von Lösungsteilen ist keine Hilfestellung, da das nicht dazu führt, ein eigenes Verständnis der Herangehensweise zu entwickeln!

Regeln

- Zwei identische Abgabeteile
  - ⇒ Eine Abgabe ist ein **Plagiat!**
  - ⇒ Das ist ein **Täuschungsversuch**
  - ⇒ Beide Abgaben gelten als nicht bearbeitet, da generell der/die Originalautor/in nicht ermittelbar ist.
- **Wiederholungsfall ⇒ Nichtbestehen – wegen Täuschung**
- **ChatGPT ⇒ Nichtbestehen – wegen Täuschung**

Identische Abgabeteile

- Abgaben werden als identisch betrachtet, wenn sie sich, u.a., nur in den
  - Variablennamen
  - Kommentaren
  - Einrückungen

  unterscheiden.

Hinweis: Wir benutzen Plagiatcheckertools!
Zusammen mit manueller Überprüfung

Acknowledgements

- Vielen Dank an:
  - Tutor*innen des Programmierkurses aus den Fachgebieten MSC und ODS

Literaturempfehlung

- Modern C, J. Gustedt
  - https://gustedt.gitlabpages.inria.fr/modern-c/
- Beej's Guide to C Programming, Brian “Beej” Hall
  - http://beej.us/guide/bgc/

Weitere Literatur

- **C**
  - Kernighan, Programmieren in C, 1990
- **Algorithmen und Datenstrukturen**
  - Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C.: Introduction to Algorithms, 3. Aufl. MIT Press Cambridge, 2009
  - Sedgewick, R.: Algorithms in C, Addison-Wesley, 2005
  - Goodrich, M. Tamassia, R.: Data Structures and Algorithms in C++, John Wiley
- **Systemsoftware**
  - Randal E. Bryant, David R. O'Hallaron „Computer Systems: A Programmer's Perspective", Prentice Hall

Ausblick

VL 0 „Organisation und Inhalt“: Ablauf der Vorlesung, Termine

VL 1 „Hello World“: „Lebenswichtiges“, Programablauf, Programmierablauf, Kompilierung und Ausführung von Programmen

VL 2 „Die ersten Schritte“: Erstes C-Programm, Elementare C-Strukturen, Datentypen, Operatoren, Schleifen

VL 3 „Kontrollstrukturen & Funktionen“: Syntax, Semantik, bedingte Anweisungen, Blöcke, Sichtbarkeit

VL 4 „Rekursive Funktionen & Bibliotheken“: rekursive Funktionsaufrufe, Modularisierung

VL 5 „Typen“: Einfache und strukturierte Datentypen, Wertebereiche, Typendefinition

VL 6 „Speicher und Adressen“: Speicher, Pointer, Funktionsaufrufe „call by value“ vs. „call by reference“

VL 7 „Speicher und Arrays“: Speicher, Arrays, mehrdimensionale Arrays, Arrays und Pointer

VL 8 „Dynamische Speicherverwaltung“: Speicherallokation, Fehlerbehandlung, Rückgabewerte, Arrays/Pointer/Adressen

VL 9 „Strings, Kanäle, Git“: Strings und Arrays, Zeichensätze, Stringlänge, Ein- und Ausgabe, Arbeiten mit git

Good luck and have a lot of fun!