Übersicht zu den Vorlesungen „Einführung in die strukturierte Programmierung“ sowie „Algorithmen und Datenstrukturen“

WI 2024 – Kurs A

Prof. Dr.-Ing. habil. Andreas Schmietendorf

Professur Wirtschaftsinformatik/Systementwicklung – HWR Berlin

Privatdozentur Software Engineering – OvG Universität Magdeburg

Abb. erstellt mit Hilfe von KI: https://playground.com/create
Promt: A computer science professor immersed in academic research … (generiert am 26.02.2024 – KI-Modell: Playground v2)

Übersicht zur Vorlesung

Motivation zur Themenstellung 3

Übersicht zu den Inhalten 3

Vorlesungstermine im Semester 5

Rahmenbedingungen und Prüfungshinweise 6

Verwendung von KI (Large Language Modelle - LLMs) 7

Literaturempfehlungen 9

Prof. Dr.-Ing. habil. Andreas Schmietendorf

Web: https://blog.hwr-berlin.de/schmietendorf/

Hochschule für Wirtschaft und Recht Berlin, FB II
Alt-Friedrichsfelde 60, D-10315 Berlin
Professur Wirtschaftsinformatik - Systementwicklung

Otto-von-Guericke-Universität Magdeburg, FIN
Universitätsplatz 2, D-39106 Magdeburg
Privatdozentur Software-Engineering

Motivation zur Themenstellung

Studierende der Wirtschaftsinformatik benötigen grundlegende Kenntnisse im Umgang mit Programmiersprachen, zu implementierenden Algorithmen als auch den in diesen Zusammenhang verwendeten Datenstrukturen. Im Diskurs der akademischen Ausbildung geht es zunächst weniger um ausschließlich eine Programmiersprache als vielmehr um die dahinterstehenden Paradigmen. Historisch wurden imperative Programmiersprachen verwendet. Mit Hilfe von nacheinander auszuführenden Befehlen werden durch den Computer umzusetzende Aufgaben festlegt. Im Kontext der Programmierung wird dabei auch von eingesetzten Kontrollstrukturen gesprochen, mit deren Hilfe der Softwareentwickler ein Programm realisiert. Konkrete Kontrollstrukturen beziehen sich auf Sequenzen, bedingte Verzeigungen, Wiederholungen von Anweisungen und den Aufruf von Unterprogrammen (Prozedur, Funktion, Methode). Beispiele für imperativ orientierte Programmiersprachen finden sich mit Cobol, Basic aber auch allen Assemblersprachen. Auch eine verhältnismäßig moderne Programmiersprache wie z.B. Java unterstützt diesen Ansatz, geht aber mit der Berücksichtigung weiterer Paradigmen massiv darüber hinaus.

Übersicht zu den Inhalten

Sämtliche Themenbereiche werden unter Verwendung der Programmiersprache Java (hier Java SE – Standard Edition) erarbeitet. Aufgrund der Ähnlichkeit zu z.B. Microsoft C# oder auch JavaScript (ist nicht Java!) sollten sich die Beispielimplementierungen leicht auf andere Systeme übertragen lassen. Entsprechend der massiv objektorientierten Ausprägung von Java (zunächst von uns nur bedingt verwendet) befindet sich der Quellcode bzw. die eingesetzten Kontrollstrukturen immer in entsprechenden Klassen!

Im Kontext des Teilbereichs „Grundlagen der strukturierten Programmierung“ werden wir uns mit den folgenden Themen auseinandersetzen:

- Einführung grundlegender Begriffe der Programmierung (Softwarelebenszyklus, Spezifikationen, Algorithmen, Syntax/Semantik, Compiler/Interpreter, elementare Algorithmen, …)
- Elementare Modellierung bzw. visuelle Darstellung von Kontrollstrukturen und Algorithmen für die strukturierte Programmierung (Struktogramme, Flussdiagramme, …)
- Codierung mit Hilfe der Programmiersprache Java (Bezeichner, Datentypen, Ein- und Ausgaben, Operatoren, Kontrollstrukturen, …)

- Grundlegende Aspekte der konstruktiven und analytischen Qualitätssicherung (ISO 25.000) und Softwaretests (strukturorientierte Testabdeckung, Unit-Tests – vgl. später JUnit - Framework zum Testen von Java-Programmen, …)
- Weiterführende Konzepte der Programmierung (Einsatz von Arrays bzw. Felder zur strukturierten Speicherung homogener Datentypen sowie Verwendung von Zeichenketten (Strings) und Methoden)

Die Inhalte des Teils „Algorithmen und Datenstrukturen werden auf die folgenden Aspekte eingehen:

- Komplexität und der Laufzeitanalyse von Algorithmen. Bestimmung der asymptotischen oberen Schranke eines Algorithmus mit Hilfe der O-Notation und praxisorientierten Benchmarking.
- Zugriff auf Filesysteme zum Lesen und Schreiben von Dateien (Verzeichnis- und Dateizugriff, Sequentieller und wahlfreier Dateizugriff)
- Implementierung dynamisch erweiterbarer Datenstrukturen. Im Details handelt es sich dabei um einfach und mehrfach verkettete Listen aber auch um sortierte Binärbäume mit und ohne AVL-Bedingung.
- Rekursive Algorithmen entsprechend des Teile-und-herrsche-Ansatzes. Rekursive Lösungen finden sich u.a. bei mathematischen Problemen (z.B. Zahlenkonvertierung), Spielen (Türme von Hanoi) oder auch in der Logistik (Jeep-Problem).
- Such und Sortieralgorithmen innherhalb unterschiedlicher Datenstrukturen bzw. unter Berücksichtigung des einhergehenden Aufwands (allg. Zeit bis zur Bereitstellung von Ergebnissen).

Vorlesungstermine im Semester

Die folgende Tabelle enthält eine grobe Zuordnung der je Veranstaltung bearbeiteten Themenstellungen. Jede Veranstaltung wird als mehrfache Kombination von Vorlesung, Übungen und Ergebnisdiskussionen (2-3 Iterationen je Termin) durchgeführt.

Termin	**SWS**	**Inhalt der Veranstaltung**
23.10.2024	6	SW-Entwicklungsprozess, Spezifikation/Algorithmen, formale Sprachen, Progr.-paradigmen, Syntax/Semantik, Compiler und Interpreter
25.10.2024	5	Motivation zur strukturierten Programmierung, Kontrollstrukturen, Struktogramme, Pseudocode, bedingte Verzweigung (if/else, switch/case), Iterationen (for, while, do/while), Entwurfsmethodiken
28.10.2024	4	Java Grundkonzepte, Eclipse-Einführung, Bezeichner, Datentypen, elementare Kontrollstrukturen (Sequenz) und Java-Operatoren
30.10.2024	4	Ein- und Ausgaben in Java, Kontrollstrukturen (if/else, switch/case, for, while, do/while), Verwendung von Unterprogrammen auf der Grundlage von Methoden
04.11.2024	6	Begriff der Softwarequalität, Lesbarkeit und Wartbarkeit von Java Programmen, Probleme im Reengineering (Legacy-Code), Softwaretest (Kontrollflussorientierte Tests, Black Box Test – vgl. JUnit)
13.11.2024	4	Felder (arrays) in Java, Einsatz von Zufallszahlen die ggf. nicht zufällig sind, Zeichenketten (Strings) in Java, Parameterübergabe in Methoden (elementare Datentypen und Referenzvariable)
15.11.2024	4	Komplexitätsbetrachtungen zur Bewertung von Algorithmen, Komplexitätsklassen entsprechend der O-Notation, Möglichkeiten zum Benchmarking unter Java
27.11.2024	4	Filverarbeitung - HW/OS unabhängiger Dateizugriff in Java (File, FileWriter, FileReader), Byteweise lesen und schreiben (FileOutputStream, FileInputStream, RandomAccessFile)
29.11.2024	6	Komplexe Datenstrukturen: einfach-, mehrfach- und zirkular verkettete Listen (dynamisch erweiterbar)
04.12.2024	4	Komplexe Datenstrukturen: sortierter und balancierter Binärbaum, Aufbauprinzip eines Binärbaums, Bedeutung der AVL-Bedingung für Suchen im Binärbaum (dynamisch erweiterbar)
06.12.2024	4	Rekursive Algorithmen (Voraussetzungen: Aktivierungsliste und Rekursionsstack, Aufrufbaum, Beispiele: Zahlenkonvertierung, Fibonacci-Reihe, Türme von Hanoi, 8-Damenproblem)
20.12.2024	5	Überblick zu Such- und Sortieralgorithmen (z.B. lin.- und bin. Suche, Hashing, Bubble-, Insert- und Merge-Sort)
06.01.2025	4	Prüfungsvorbereitung (allgemeine Fragen, Reflektion der Probeklausur, Prüfungsschwerpunkte)

Rahmenbedingungen und Prüfungshinweise

Die Bereitstellung sämtlicher Unterlagen zur Vorlesung erfolgt über die an der HWR Berlin als Standard eingesetzte Lehrplattform Moodle (https://moodle.hwr-berlin.de). Im Detail handelt es sich dabei um Vorlesungsskripte, integrierte Übungsaufgaben, erläuterte Lösungsansätze und korrespondierende Quellcodefragmente. Darüber hinaus erfolgt die Vorstellung von ausgewählten Programmieraspekten über den Einsatz kurzer Lehrvideos, welche ebenfalls über Moodle bereitgestellt werden. Für die Wiedergabe dieser im mp4-Format vorliegenden Videos wird der VLC Media Player empfohlen (da dort getestet), alternative Werkzeuge sollten allerdings auch einsetzbar sein. Eine Weitergabe der bereitgestellten Unterlagen über die Grenzen der HWR Berlin ist ausdrücklich untersagt, darüber hinaus ist das Aufzeichnen von Vorlesungen nur in Abstimmung mit den jeweiligen Dozenten erlaubt. Weiterführende Informationen zur vertretenen Professur finden sich unter https://blog.hwr-berlin.de/schmietendorf, ebenso die Möglichkeiten zur Kontaktaufnahme.

Für den erfolgreichen Abschluss ist die Nutzung eines eigenen Computers bzw. Laptops und einer darauf installierten Eclipse-Entwicklungsumgebung (kann theoretisch vom USB-Stick gestartet werden) unabdingbar! Entsprechend den Laborbedingungen der HWR Berlin (speziell Fachrichtung Wirtschaftsinformatik am FB2) wird eine Windows-Installation unterstellt. Unter anderen Betriebssystemen wie z.B. macOS sollten die Beispiele allerdings ebenfalls lauffähig sein, ggf. sind kleinere Anpassungen (u.a. Aspekte des eingesetzten Filesystems) notwendig. Sämtliche Beispiele sollten maximal unter Verwendung des JavaSE Compliance-Level 14 (nicht höher) implementiert bzw. ausgetestet werden. Grundsätzlich werden zu allen der ca. 50 Übungen entsprechende Lösungsansätze (d.h. Java-Quellcode) bzw. detaillierte Übungsblätter bereitgestellt, zumeist allerdings erst nachdem durch die Studierenden eigene Implementierungen realisiert bzw. Erfahrungen gesammelt wurden. Darüber hinaus wird von den Studierenden die Vorstellung eigener Lösungsansätze (d.h. mind. eine Präsentation ist je Student/-in im Semester nachzuweisen) im Rahmen der Vorlesungstermine erwartet! Abgeschlossen wird das Fach durch eine 2-stündige Klausur (jeweils 5 Fragenkomplexe). Im laufenden Semester (ca. Mitte Dezember) wird eine entsprechende Probeklausur zur Verfügung gestellt.

Hinweis: Im Rahmen der Vorlesungen können nicht alles Aspekte des behandelten Stoffs angesprochen werden. Dem entsprechend wird eine intensive Ausei-

nandersetzung mit den bereitgestellten Unterlagen bzw. das Hinzuziehen weiterführender Literatur im Selbststudium erwartet! Darüber hinaus wird die Bildung von Lerngruppen ausdrücklich empfohlen!

Verwendung von KI (Large Language Modelle - LLMs)

Die Verwendung von KI (hier speziell Modelle der generativen KI – primär große Sprachmodelle wie z.B. OpenAI bzw. ChatGPT) zur Unterstützung programmiertechnischer Tätigkeiten ist ausdrücklich erwünscht! Grundsätzlich sind die bereitgestellten Ergebnisse aus inhaltlicher Sicht, den ggf. einhergehenden Risiken aber auch aus Sicht der Urheberschaft kritisch zu hinterfragen. Darüber hinaus ist zu berücksichtigen, dass derartige Ergebnisse nicht zitierfähig sind! Mit Hilfe von Anlagen zum originären Quellcode lassen sich KI-Ergebnisse gleichwohl sinnvoll dokumentieren. Auf dieser Grundlage lassen sich die im wissenschaftlichen Kontext existierenden Forderungen nach Transparenz und Reproduzierbarkeit bedingt gewährleisten.

An der HWR Berlin wird ein datenschutzkonformer KI-Zugang (Service "HAWKI" – Basis: ChatGPT 4 (Open AI)) unter der folgenden URL bereitgestellt:

https://blog.hwr-berlin.de/elerner/kuenstliche-intelligenz/

Die folgende Anfrage wurde am 15. Oktober 2024 an den HAWKI-Services der HWR Berlin gesendet. Unterhalb der Abbildung kann die KI-generierte Antwort (durchgängig kursiv) eingesehen werden.

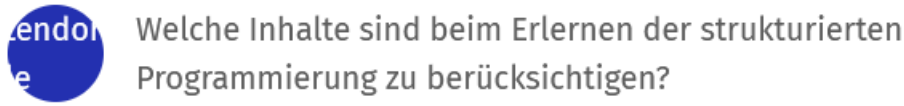

Abbildung 0-1: Exemplarische Anfrage an den HAWKI-Service der HWR Berlin

Beim Erlernen der strukturierten Programmierung sind mehrere grundlegende Konzepte und Prinzipien zu berücksichtigen. Diese bilden die Basis für das Schreiben klarer, verständlicher und wartbarer Programme. Zu den wesentlichen Inhalten gehören:

1. *Grundlegende Kontrollstrukturen: - Sequenz: Die Ausführung von Anweisungen in der Reihenfolge, in der sie im Code erscheinen. - Selektion (Bedingte Anweisungen): Entscheidungsstrukturen wie `if`, `else if`, `else`, und `switch`, die es ermöglichen, unterschiedliche Pfade basierend auf*

Bedingungen auszuwählen. - Iteration (Schleifen): Wiederholungsstrukturen wie `for`, `while`, und `do-while`, die es erlauben, Codeblöcke mehrere Male auszuführen.

2. *Modularität: - Funktionen und Prozeduren: Das Zerlegen eines Programms in kleinere, unabhängige Einheiten (Funktionen oder Methoden), die spezifische Aufgaben erledigen. Dies erleichtert die Wartung und Wiederverwendbarkeit von Code. - Parameterübergabe und Rückgabewerte: Wie Funktionen Daten entgegennehmen und Ergebnisse zurückgeben.*
3. *Datenstrukturen: - Primitive Datentypen: Integer, Float, Char, Boolean, etc. - Komplexere Strukturen: Arrays, Listen, Strukturen (Structs) und andere Sammlungstypen.*
4. *Lesbarkeit und Wartbarkeit: - Code-Kommentierung: Nutzung von Kommentaren, um den Zweck und die Funktionsweise von Codeabschnitten zu beschreiben. - Namensgebung: Klarheit und Aussagekraft bei der Benennung von Variablen und Funktionen. - Formatierung: Konsistentes Einrücken und geeignete Verwendung von Leerzeichen, um den Code übersichtlich zu gestalten.*
5. *Debugging und Fehlersuche: - Grundlegende Techniken: Verwenden von Debuggern oder Ausgabestatements, um Programmfehler zu identifizieren und zu beheben. - Fehlerarten: Verständnis von Syntaxfehlern, Laufzeitfehlern und logischen Fehlern.*
6. *Dokumentation: - Erstellen von Benutzerdokumentation: Umfassende Beschreibung der Funktionen und der Nutzung des Programms. - Kommentare im Code: Inhaltliche Ergänzungen zur besseren Lesbarkeit und Wartbarkeit. Diese Aspekte der strukturierten Programmierung führen zu effizientem Code, der leichter zu verstehen, zu testen und zu warten ist. Beim Lernen sollten praktische Übungsaufgaben integriert werden, um das neu erworbene Wissen aktiv anzuwenden.*

Die KI-genereiten Inhalte finden sich nahezu exakt in der hier beschriebenen Vorlesung wieder. Lediglich fortgeschrittenen Datenstrukturen wie der Einsatz von Files bzw. Filesystemen oder auch Binärbäume werden nicht direkt angesprochen. Außen vor bleiben ebenso rekursive Algorithmen bzw. die ebenfalls in die Vorlesung aufgenommenen Such- und Sortieralgorithmen.

Literaturempfehlungen

Krüger, G.; Stark, T.: Handbuch der Java-Programmierung – Version 6, München, u.a.: Addison-Wesley, 2013 (download www.heise.de)

Petri, Björn; Petri, Britta: Java-Tutorial.org - Java lernen leicht gemacht (Java SE Tutorial), (https://www.java-tutorial.org/index.html), letzter Zugriff: Oktober 2019

Gumm, H. P.; Sommer, M.: Einführung in die Informatik, 8. Auflage. München, Oldenbourg Verlag, 2009 (ISBN 978-3-486-58724-1)

Sanchez, J.; Canton, M. P.: Java 2 – Wochenend Crashkurs, 4. Auflage, mitp-Verlag, 2001 (ISBN 3-8266-0769-4) – leider nicht mehr verlegt

Sedgewick, R.: Algorithmen in Java: Pearson Studium – Addison Wesley, 2003 (ISBN 3-8273-7072-8)

Kopec, D.: Algorithmen in Java - 32 Klassiker der Informatik, von Rucksackproblem bis Neuronale Netze, Rheinwerk Verlag, 2021 (ISBN 978-3-8362-8452-3)