Informatik II

Oliver Jack

Fachhochschule Jena
Fachbereich Elektrotechnik und Informationstechnik

Sommersemester 2010

Vorlesung 1. Organisatorisches und Einführung

Organisatorisches

- Organisation der Veranstaltung
- Inhalt der Veranstaltung
- Literatur und Werkzeuge

Einführung

- Informatik
- Algorithmus

Zusammenfassung

1. Organisatorisches und Einführung

Heutige Vorlesung

- Organisatorisches
- Gebiete der Informatik
- Algorithmus

Lernziele dieser Vorlesung

- Kenntnis der Inhalte dieser Lehrveranstaltung
- Kenntnis des prinzipiellen Aufbaus und der prinzipiellen Eigenschaften eines Algorithmus

Organisatorisches

- Studiengang: ET/ME (BA)
- Dozent: Prof. Dr.-Ing. Oliver Jack
- Turnus
  - V: wöchentlich eine Doppelstunde, Mittwochs 07:45 bis 09:15 Uhr
  - Ü: 14-tägig eine Doppelstunde gemäß zentraler Stundenplanung
  - P: wöchentlich eine Doppelstunde gemäß zentraler Stundenplanung
- Aktuelle Terminänderungen zentral über `http://stundenplanung.fh-jena.de`
- Ort: 04.00.02(HS7)
- Skript: nein
- Folien: aktuelle Folien nachschüssig (über Website)
  Es wird ein Passwort benötigt, das in der Lehrveranstaltung bekanntgegeben wird.

Organisatorisches (Forts.)

Kommuniukation

- Prof. Dr.-Ing. Oliver Jack
- Tel. 03641/205-715
- E-Mail: Oliver.Jack@fh-jena.de
- Website: http://www.et.fh-jena.de/Fachber/Visitenkarten/Jack.htm
  $\longrightarrow$ *Homepage*
- Büro: 05.02.09
- Sprechzeiten: Mittwochs 09:30–11:00 Uhr
- Aktuelle Informationen zur Veranstaltung auf der Website

Übungen

- Dozenent: Herr Berger, Herr Fiedler
- Gruppen gemäß Set-Einteilung
- Termine zentral über `http://stundenplanung.fh-jena.de`
- Übungen finden im Rechnerpool im Grundlagenbereich statt, Raum 01.01.11
- Es wird ein Hochschulpasswort benötigt

Praktika

- Dozenten: Prof. Jack
- Gruppen gemäß Set-Einteilung
- Termine zentral über `http://stundenplanung.fh-jena.de`
- Praktikum findet im Rechnerpool im Grundlagenbereich statt, Raum 01.01.11
- Es wird ein Hochschulpasswort benötigt

Praktische Übungen im Selbststudium

- An der FH: im Rechnerpool
- Zu Hause: irgendein C++ Compiler

Didaktischer Rahmen

- Vorlesung: Vermittlung der theoretischen Grundlagen
- Übung: Eigenständige Rekapitulation der Theorie, (mitunter längeres) Nachdenken zur Lösung (mitunter schwieriger) Aufgaben, Kleingruppenarbeit empfohlen

Modulprüfung

- Alternative Prüfungsleistung
  - Software-Entwicklungsprojekt / Programmierprojekt im Rahmen der Übungen
  - Bearbeitung in Kleingruppen gemäß Set-Einteilung
  - Präsentation des Ergebnisses in den letzten Übungen der Vorlesungszeit
  - Jedes Gruppenmitglied präsentiert einen Teil des Ergebnisses
- Wiederholungsmöglichkeit: nach 2 Semestern
- Achtung: Note ist Bestandteil des Abschlusszeugnisses!

Vorlesung

Wesentliche Inhalte

- Datenstrukturen und Algorithmen (exemplarisch)
- Objektorientierte Programmierung in C++ (Polymorphie, Klassen, Verebung, Schablonen)
- Ausnahmebehandlung, Fehlertoleranz
- Ein wenig Software-Technik (Entwurf, Programmierung)

Präsentation

- Folien und gelegentlich Tafel

Erwartungen

- Aktive Teilnahme an Vorlesung und Übung
- Aktives eigenständiges Programmieren
- Auseinandersetzen mit den Übungsaufgaben, zumindest Fragen formulieren können
- Gemeinsam Spaß beim Studium von C++ haben

Anregungen zur Veranstaltung

- Vorbereiten in Kleingruppen empfehlenswert
  - Miteinander lernen
  - Reihum Tutor/Prüfer spielen
- In der Sache
  - Revidiertes schriftliches Material durcharbeiten
  - Erst in die Breite, dann in die Tiefe lernen:
    - Überblick bekommen, Zusammenhänge erkennen
    - Dabei auch die Details beherrschen lernen
  - Beispiele zu allen wesentlichen Begriffen zurecht legen
  - Üben, sich in der Fachsprache auszudrücken
  - Üben, die Formalismen zu benutzen

Literatur


Bjarne Stroustrup.
Die C++-Programmiersprache.
Addison-Wesley, Boston et al, Deutsche Übersetzung der Special edition, 2000.


Scott Meyers.
Effektiv C++ programmieren.
Addison-Wesley, Boston et al, 2005.


Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo.
C++ Primer.
Addison-Wesley, Boston et al, 4. edition, 2006.

Literatur


Dietrich May.
Grundkurs Software-Entwicklung mit C++.
Vieweg, Wiesbaden, 2. edition, 2003.


Robert Sedgewick.
Algorithmen in C++, Band 1. Teile 1–4: Grundlagen, Datenstrukturen, Sortieren, Suchen.
Pearson Studium, München et al, 3. edition, 2002.

Entwicklungsumgebungen

Linux

- KDevelop
- Emacs (sehr mächtig, etwas gewöhnungsbedürftig)
- Eclipse (Ganymede oder Galileo) mit C/C++ Development Tooling (CDT)

Entwicklungsumgebungen (Forts.)

Windows

- Code::Blocks (das einfachste, empfohlen)
- MinGW (Compiler, im wesentlichen gcc) + gdb (Debugger) + Emacs
- Cygwin/X (Linux unter Windows, enthält gcc + gdb + Emacs)
- Eclipse (Ganymede oder Galileo) mit C/C++ Development Tooling (CDT) zusammen mit MinGW + gdb oder Cygwin
- Kommerzielle (Microsoft Visual C++ / Visual Studio, Borland C++-Builder)

Entwicklungsumgebungen (Forts.)

Mac OS X

- XCode
- gcc + gdb + Emacs
- Kommerzielle (CodeWarrior, ...)

Wieso sitzen Sie hier? (Dozentensicht)

Gundlagen lernen

- Begriffe, Denkweisen
- Programmierung
- Etwas Software-Technik

Spaß haben

- Teamarbeit
- Computer beherrschen

Wieso sitzen Sie hier? (Studentensicht)

Schein, Prüfungsleistung

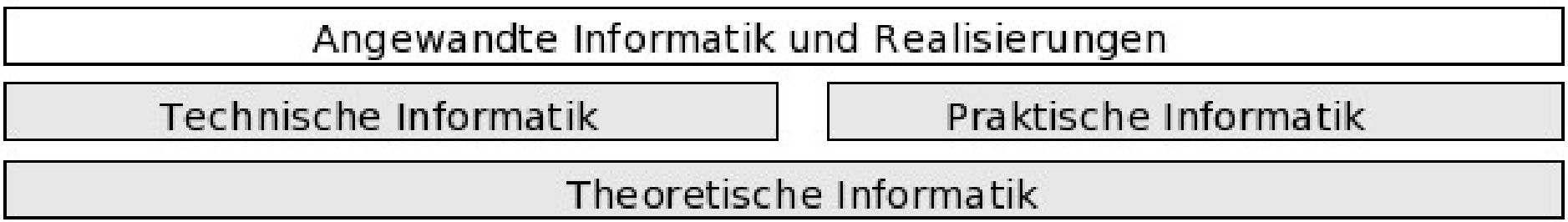
- Aufwand: wie Informatik I
- Aber: Alternative Prüfungsleistung, Gruppenarbeit, Präsentation, Semesterprojekt

Was ist Informatik? (engl. Computer Science)

Kunstwort aus Information und Automatik

oder aus Information und Mathematik. Wissenschaft von der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, besonders der automatischen Verarbeitung mit Hilfe von Computern.

Gebiete der Informatik



Informatik II: Praktische Informatik

Was ist Information?

Definitionsversuch

Information ist ein potenziell oder tatsächlich vorhandenes nutzbares oder genutztes Muster von Materie und/oder Energieformen, das für einen Betrachter innerhalb eines bestimmten Kontextes relevant ist.
Wesentlich für die Information ist die Wiedererkennbarkeit sowie der Neuigkeitsgehalt.
Das verwendete Muster verändert den Zustand eines Betrachters im menschlichen Zusammenhang insbesondere dessen Wissen.
Formaler ist Information die Beseitigung von Unbestimmtheit.

Was ist ein Algorithmus?

Algorithmus: (anschaulich)

- Beschreibung eines Weges vom Problem zur Lösung in endlich vielen Schritten.
- Randbedingungen:
  1. Der Weg muss formal so präzise definiert sein, dass er im Prinzip von einer Maschine (rein mechanisch) gegangen werden kann.
  2. Problem und Lösung müssen vorher formal spezifiziert werden.

Algorithmus: Formal

- Ein Algorithmus gibt an, wie Eingabedaten schrittweise in Ausgabedaten umgewandelt werden.
- Er beschreibt also eine Abbildung

$$f : E \rightarrow A$$

  von der Menge der Eingabedaten E in die Menge der Ausgabedaten A und wie die Abbildung zu "berechnen" ist.
- Ein Algorithmus wird korrekt genannt, wenn er
  1. den spezifizierten Zusammenhang zwischen E und A für alle Eingaben aus E erfüllt und wenn er
  2. terminiert.

Algorithmus: Beispiel "Finde jüngste Person hier im Raum"

Problemanalyse

- Annahme: Es sind $n \geq 1$ Personen im Raum
- Formulierung "jüngste Person" eindeutig? Nein!
  1. Genauigkeit der Altersangabe in Sekunden oder Tage oder Jahre?
  2. Es könnten ≥ 2 Personen gleichen Alters im Raum sein!

Spezifikation

- Gegeben: Folge von $n \geq 1$ Altersangaben $a_1, a_2, \ldots, a_n$ in Jahren
- Gesucht: $a_j = \min(a_1, a_2, \ldots, a_n)$, wobei j die erste Stelle in der Folge sei, an der das Minimum auftritt

Algorithmus: Beispiel “Finde jüngste Person hier im Raum” (Forts.)

Algorithmenentwurf, Vor- und Nachbedingungen

- Gegeben: Folge von n Altersangaben $a_1, a_2, \ldots, a_n$ in Jahren, $n \geq 1$
- Gesucht: $a_j = \min(a_1, a_2, \ldots, a_n)$, wobei j die erste Stelle in der Folge sei, an der das Minimum auftritt

Algorithmus: Beispiel “Finde jüngste Person hier im Raum” (Forts.)

Algorithmenentwurf, Schrittfolge

(1) [Wähle 1. Kandidat] Setze $j = 1$ und $x = a_j$;
(2) [Suchlauf]
    Setze $i = 2$.
    Solange $i \leq n$ gilt,
        falls $a_i < x$, dann setze $j = i$ und $x = a_j$;
        [jetzt gilt $a_j = \min(a_1, \ldots, a_i)$]
        erhöhe i um 1
(3) [Ausgabe] Person j mit Alter x ist eine jüngste Person

Algorithmus: Beispiel “Finde jüngste Person hier im Raum” (Forts.)

Gegeben: Folge von 8 Altersangaben 20, 21, 20, 19, 18, 19, 18, 20

	1	2	3	4	5	6	7	8	j
(1)	20								1
(2)	20	21							1
(2)	20	21	20						1
(2)	20	21	20	19					4
(2)	20	21	20	19	18				5
(2)	20	21	20	19	18	19			5
(2)	20	21	20	19	18	19	18		5
(2)	20	21	20	19	18	19	18	20	5

Algorithmus: Beispiel "Finde jüngste Person hier im Raum" (Forts.)

Korrektheit

- Behauptung: Der Algorithmus ist korrekt.
- Beweis: Wenn der Algorithmus anhält, dann ist
  1. $a_j = \min(a_1, \ldots, a_i)$ mit $1 \leq i \leq n$.
     Das gilt für $i = 1$ nach Schritt (1) und während des Suchlaufs invariant für alle i an der angegeben Stelle.
  2. j ist die erste Stelle, an der 1. gilt, weil im Fall $a_i = x$ kein Austausch mehr stattfindet, sondern nur bei $a_i < x$.

  Der Algorithmus hält an, nachdem $i = n$ war

.

Effizienz

Wir messen den Zeitaufwand in Einheiten E.

Aktion	Aufwand	Häufigkeit der Aktion
Setze $j = 1, x = a_j$	2 E	1
Setze $i = 2$	1 E	1
Test $i = n$	1 E	n
Test $a_i < x$	1 E	$n - 1$
Setze $j = i, x = a_j$	2 E	A
Erhöhe i	1 E	$n - 1$

Insgesamt also:

$$T(n) = 2 + 1 + n + (n - 1) + 2A + (n - 1)E = (3n + 2A + 1)E$$

Effizienz (Forts.)

- $T(n) = (3n + 2A + 1)E$
- Welche Werte kann A annehmen?
- Hier: zwei Szenarien
  1. Schlimmster Fall (engl. worstcase): $A = n - 1$ d.h., das Alter aller Personen ist paarweise verschieden und es ist in der Aufzählung absteigend sortiert
     $$T_{max}(n) = (5n - 1)E$$
  2. Bester Fall (engl. best case): $A = 0$ d.h., erste Person in der Aufzählung ist bereits die jüngste Person
     $$T_{min}(n) = (3n + 1)E$$

Zusammenfassung

- Lernen erfolgt in Kleingruppenarbeit
- Ein Semesterprojekt wird bearbeitet
- Diese Lehrveranstaltung behandelt Themen aus der praktischen Informatik.
- Ein Algorithmus entsteht durch Problemanalyse, Spezifikation, Entwurf der Schrittfolge mit Vor- und Nachbedingungen.
- Wichtige Eigenschaften eines Algorithmus sind Korrektheit und Effizienz.