

C Grundlagen

- Programmierung
- Programmerstellung und -ausführung
- Organisation eines Rechners
 - ◆ Klassischer Universalrechner
 - ◆ Datendarstellung im Rechner
 - ◆ Dateien
 - ◆ Benutzerumgebung
 - ◆ Programmierumgebung
- Begriffe

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.5

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.2 Programm

- Daten: Abstrakte Darstellung des problemrelevanten Teils der realen Welt
- Algorithmus: Schrittfolge von Operationen auf den Daten, die als allgemeines Verfahren die Lösung einer Klasse von Problemen zum Ziel hat
 - endliche Beschreibung
 - eindeutige Operationsfolge
es steht stets fest, welches der nächste Schritt ist
 - Problemklasse: z. B. größter gemeinsamer Teil von zwei beliebigen (!) natürlichen Zahlen

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.7

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.1 Programmierung

- Einsatz von Rechenanlagen
 - ◆ Informationen speichern, strukturieren, sortieren, zugänglich machen
 - ◆ Rechnen an sich ist eher selten
- Problemlösung mit Rechenanlagen
 - ◆ Programmierung

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.6

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.3 Beispiel (1) — Polynomrechnung

- Problemklasse: Berechnung von $f(x) = ax^3 + bx^2 + cx + d$
- Daten: Parameter $a, b, c, d, zw_{1,2,3,4} \in \mathbb{R}$
- Algorithmus: Horner-Schema

$$\begin{array}{l} a \boxed{} * x \boxed{} = zw_1 \boxed{} \\ \phantom{a \boxed{} * x \boxed{}} + b \boxed{} \\ \phantom{a \boxed{} * x \boxed{}} = zw_2 \boxed{} \\ \phantom{a \boxed{} * x \boxed{}} \downarrow \\ zw_2 \boxed{} * x \boxed{} = zw_3 \boxed{} \\ \phantom{zw_2 \boxed{} * x \boxed{}} + c \boxed{} \\ \phantom{zw_2 \boxed{} * x \boxed{}} = zw_4 \boxed{} \\ \phantom{zw_2 \boxed{} * x \boxed{}} \downarrow \\ zw_4 \boxed{} * x \boxed{} = zw_5 \boxed{} \\ \phantom{zw_4 \boxed{} * x \boxed{}} + d \boxed{} \\ \phantom{zw_4 \boxed{} * x \boxed{}} = y \boxed{} \end{array}$$

Schritte:

1. $a * x \rightarrow zw_1$
2. $zw_1 + b \rightarrow zw_2$
3. $zw_2 * x \rightarrow zw_3$
4. $zw_3 + c \rightarrow zw_4$
5. $zw_4 * x \rightarrow zw_5$
6. $zw_5 + d \rightarrow y$

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.8

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.4 Beispiel (1) — Programm

```
main()
{
    /* Beschreibung der Daten */
    float x;          /* Argument zu dem Pol. berechnet wird */
    float y;          /* Polynomwert */

    float a = 11, b = 7, c = 5, d = 3;
    float zw1, zw2, zw3, zw4, zw5;    /* Zwischenergebnisse */

    /* Einlesen des Arguments */
    printf("Argument = ");
    scanf("%f", &x);

    /* Kontrollausgabe der Eingabedaten */
    printf("Der Polynomwert soll fuer das");
    printf(" Argument %f berechnet werden.\n", x);

    /* Berechnung des Polynomwertes mit HornerSchema */
    zw1 = a * x;
    zw2 = zw1 + b;
    zw3 = zw2 * x;
    zw4 = zw3 + c;
    zw5 = zw4 * x;
    y = zw5 + d;

    /* Ausgabe des Ergebnisses */
    printf("Ergebnis: f(%f) = %f\n", x, y);
}
```

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

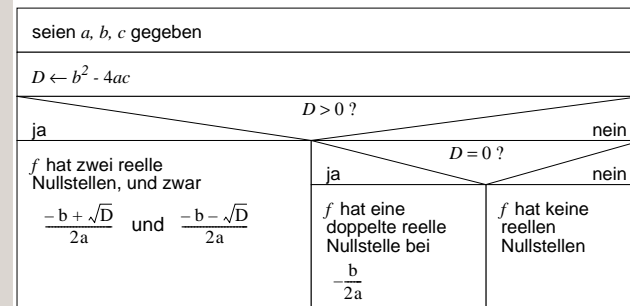
C.9

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.6 Beispiel (2) — Struktogramm

- einfache Dokumentation des Programmentwurfs

Nullstellen — bestimmt reelle Nullstellen von $f(x) = ax^2 + bx + c$



C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.11

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.5 Beispiel (2) — Nullstellen einer Funktion

- Problemklasse: Was sind die reellen Nullstellen der Funktion $f: \mathbb{R} \rightarrow \mathbb{R}$ mit $f(x) = ax^2 + bx + c$
- Daten: Parameter $a, b, c \in \mathbb{R}$, Diskriminante D
- Algorithmus:
 1. seien a, b, c gegeben
 2. berechne die Diskriminante D zu $b^2 - 4ac$
 3. wenn $D > 0$ weiter mit Schritt 7
 4. wenn $D = 0$ weiter mit Schritt 9
 5. f hat keine reellen Nullstellen
 6. fertig
 7. f hat zwei reelle Nullstellen, und zwar $\frac{-b + \sqrt{D}}{2a}$ und $\frac{-b - \sqrt{D}}{2a}$
 8. fertig
 9. f hat eine doppelte reelle Nullstelle bei $-\frac{b}{2a}$
 10. fertig

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.10

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.7 Beispiel (2) — Programm

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

main ()
{
    float a, b, c;          /* Parameter in a*x^2+b*x+c */
    float dis;             /* Diskriminante */

    /* Initialisierung: Einlesen der Parameter */
    printf("Nullstellen von f(x)=a*x^2+b*x+c\n");
    printf("a=");
    scanf("%f", &a);
    printf("b=");
    scanf("%f", &b);
    printf("c=");
    scanf("%f", &c);

    dis = b*b - 4.0 * a * c;    /* Berechnung der Diskriminante */

    /* Berechnung der Nullstellen */
    if ( dis == 0.0 )          /* doppelte Nullstelle */
        printf("Doppelte Nullstelle bei %f\n", -b/(2.0*a));
    else if ( dis > 0.0 ) {   /* zwei reelle Nullstellen */
        printf("1. Nullstelle bei %f\n", (sqrt(dis)-b)/(2.0*a));
        printf("2. Nullstelle bei %f\n", (-sqrt(dis)-b)/(2.0*a));
    }
    else                     /* keine reelle Nullstelle */
        printf("Keine reelle Nullstelle\n");
}
```

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

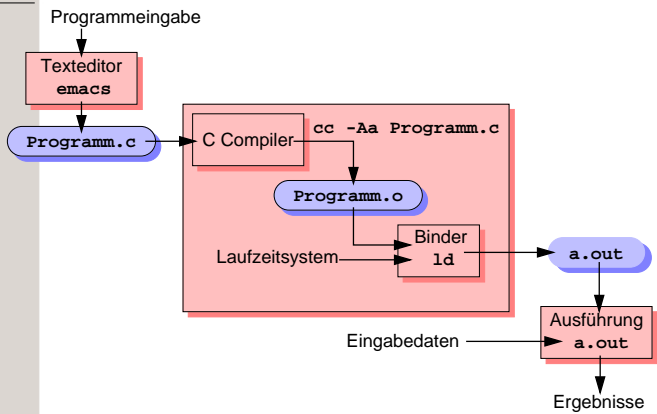
C-Grundlagen.doc 1998-05-15 17.37

C.12

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.8 Programmiererstellung und -ausführung

1 Vorgehensweise



3 Übersetzung und Ausführung der Beispielprogramme

C.8 Programmiererstellung und -ausführung

Ausführung des Beispiels 'Polynom'

```
faii06a% cc -Aa polynom.c
faii06a% a.out
Argument = 4
Der Polynomwert soll fuer das Argument 4.000000 berechnet werden.
Ergebnis: f(4.000000) = 839.000000
faii06a%
```

Ausführung des Beispiels 'Nullstellen'

```
faii06a% cc -Aa Nullstellen.c -lm
faii06a% a.out
Nullstellen von f(x)=a*x^2+b*x+c
a=1
b=2
c=-3
1. Nullstelle bei 1.000000
2. Nullstelle bei -3.000000
faii06a%
```

2 einfaches Beispiel

C.8 Programmiererstellung und -ausführung

■ Programm

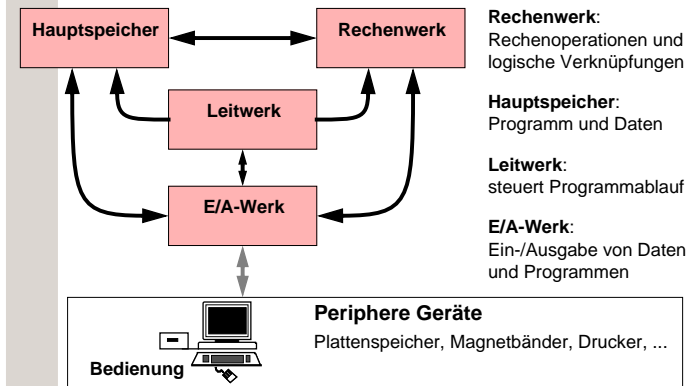
```
main()
{
    printf("hello world\n");
}
```

■ Übersetzung und Ausführung

```
faii06a% cc hello.c
faii06a% a.out
hello world
faii06a%
```

C.9 Klassischer Universalrechner

■ Burks, Goldstine und von Neumann (Princeton 1946/47)



Rechenwerk:
Rechenoperationen und logische Verknüpfungen

Hauptspeicher:
Programm und Daten

Leitwerk:
steuert Programmablauf

E/A-Werk:
Ein-/Ausgabe von Daten und Programmen

Periphere Geräte
Plattenspeicher, Magnetbänder, Drucker, ...

C.10 Datendarstellung im Rechner

- Alle Informationen (Daten, Texte, Programmanweisungen) werden im Speicher durch Bitfolgen dargestellt
 - ◆ **Bit** = Speicherstelle mit Wert 0 oder 1
 - ◆ Beispiele: Zahl 2 = 10, Zahl 5 = 101, Buchstabe a = Zahl 97 = 1100 0001
- 8 Bit bilden ein **Byte**
- Für ganze Zahlen werden meist 32 Bit vorgesehen (ein **Wort**)
- Zeichen (Buchstaben) werden durch Zahlen kodiert
- Ein Wort ist normalerweise die Einheit, mit der das Rechenwerk operiert
- Das Rechenwerk führt entsprechend den Anweisungen des Programms arithmetische oder logische Verknüpfungen von Worten durch
- Die Worte auf denen das Rechenwerk operiert, werden in **Registern** des Rechenwerks zwischengespeichert (für schnelleren Zugriff)

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.17

Reproduktion jeder Art oder Vervielfältigung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Struktur des Dateisystems

C.11 Dateien und Dateisystem

- Hierarchische Strukturierung des Dateisystems
 - ◆ reguläre Dateien
 - Können beliebige Daten speichern (Texte, Zahlen, Programme, ...)
 - Einfache Bedienung (öffnen, lesen, schreiben)
 - ◆ Dateikataloge (*Directories*)
 - Enthalten reguläre Dateien oder weitere Kataloge
- ➔ Baumförmige Struktur des Dateisystems
- ◆ Dateien werden durch Aneinanderreihung der Katalognamen und des Dateinamens eindeutig benannt

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.19

Reproduktion jeder Art oder Vervielfältigung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.11 Dateien und Dateisystem

1 Allgemeines

- Dateien sind abstrakte Gebilde zur Speicherung von Daten auf einem Hintergrundspeicher (Festplatte, Diskette)
 - ◆ Verbergen die reale Struktur (Sektoren, Zylinder, etc.)
 - ◆ Haben einen Namen (Zeichenkette)
 - ◆ UNIX verwaltet Zugriffsrechte (welcher Benutzer darf lesen, schreiben oder ein Programm in der Datei ausführen)

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

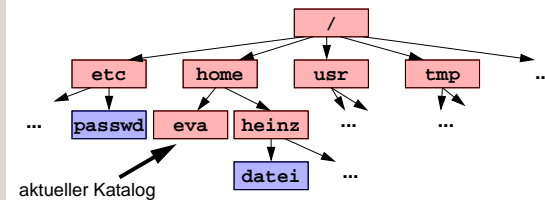
C.18

Reproduktion jeder Art oder Vervielfältigung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

3 Pfade im Dateisystem

C.11 Dateien und Dateisystem

- Baumstruktur & Pfade im Dateibaum



- ◆ z. B. `/home/heinz/datei` `/tmp` `../heinz/datei`
- ◆ `/` ist Trennsymbol (*Slash*) zwischen Pfadkomponenten
Pfade, die mit `/` beginnen, starten im Wurzelkatalog (*root*)
sonst Beginn implizit im aktuellem Katalog
- ◆ jeder Katalog enthält Verweise auf sich selbst (`.`) und auf den darüberliegenden Katalog (`..`) — in Pfadnamen verwendbar!
- ◆ aktueller Katalog kann mit Kommando `cd` gewechselt werden
z. B. `cd ../heinz`

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.20

Reproduktion jeder Art oder Vervielfältigung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.12 Benutzerumgebung

- Das Betriebssystem (UNIX) und eine Reihe von Dienstprogrammen (*Tools*) stellen eine Umgebung bereit, die von der eigentlichen Rechnerarchitektur abstrahiert
- Jedem Benutzer ist ein eigener Dateikatalog zugeordnet (*Home-Directory*)
- Nach dem *Login* wird automatisch eine Fensteroberfläche mit mehreren Fenstern gestartet (*X-Windows*)
 - ◆ Weitere Fenster sind über Menüs mit der Maus zu starten
- Jedes Fenster entspricht einem virtuellen Terminal
 - ◆ Kommandointerpreter (Shell) zur Eingabe von Kommandos (C-Compiler, Dateikatalog auflisten, ...)
 - ◆ Editor (Emacs) zur Eingabe von Programmen und Texten
 - ◆ ...

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

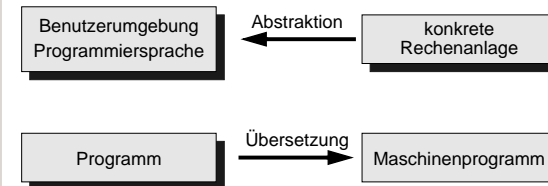
C-Grundlagen.doc 1998-05-15 17.37

C.21

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.13 Programmierumgebung (2)

- Abbildung Programmiersprache → Maschinensprache
 - ◆ Interpreter: Ein Dienstprogramm interpretiert das Programm des Anwenders
 - ◆ Compiler: Ein Dienstprogramm übersetzt das Anwenderprogramm in Maschinensprache



C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.23

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.12 Benutzerumgebung (2)

- Beschreibung über die Benutzerumgebung und alle Kommandos sind online auf dem Rechner abrufbar
 - ◆ Integriert in World-Wide-Web (WWW)
 - ◆ Tool: *Mosaic* (über Menü starten, dann selbsterklärend)

C.13 Programmierumgebung

- Programmiersprache bietet Sicht auf eine abstrakte Rechenanlage
 - ◆ Statt Bits, Bytes und Worten:
 Ganze und reelle Zahlen, Zeichen
 - ◆ Statt einfachen Abfragen und direkten Sprüngen an Adressen:
 if - then - else, Schleifen, ...

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.22

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

C.14 Begriffe

Programm:

Eine in einer Programmiersprache abgefaßte Verarbeitungsvorschrift mit einer (evtl. impliziten) Vereinbarung der Daten

Maschinensprache:

Programmiersprache, die nur Anweisungen zuläßt, die von einer bestimmten Rechenanlage (unmittelbar) ausgeführt werden können

Maschinenprogramm:

Ein in Maschinensprache abgefaßtes Programm

Maschinenorientierte Programmiersprache:

Programmiersprache, deren Anweisungen ähnliche/gleiche Struktur haben wie die der Maschinensprache einer bestimmten Rechenanlage

Maschinenorientiertes Programm:

Ein in maschinenorientierter Programmiersprache abgefaßtes Programm

Höhere / problemorientierte Programmiersprache:

Programmiersprache, die dazu dient, Programme eines bestimmten Anwendungsbereiches unabhängig von einer bestimmten Rechenanlage abzufassen, und die dem Anwendungsbereich besonders angemessen ist (FORTRAN, PASCAL, COBOL, C, C++, ...).

C - Programm:

Ein in der Programmiersprache C abgefaßtes Programm

C-Ing

Einführung in die Programmierung für Ingenieure — C
© Jürgen Kleinöder, Universität Erlangen-Nürnberg, IMMD IV, 1998

C-Grundlagen.doc 1998-05-15 17.37

C.24

Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.