

Übersicht

Lehrinhalt: Programmierung in C

- **Überblick über Programmiersprachen, Allgemeines**
- C-Basisdatentypen, Zahlendarstellung, Variablen, Konstanten
- Operatoren und Ausdrücke
- Anweisungen
- Kontrollstrukturen
- Funktionen
- Zeiger und Felder
- Zeichenketten (Strings)
- Benutzerdefinierte Datentypen
- Dynamischer Speicher
- Dateiarbeit
- Funktionspointer, Rekursion
- Preprozessor



Allgemeines

- Verschiedene Sprachkonzepte
- C-Sprachfamilie
- C-ähnliche Programmiersprachen
- Allgemeines zu C

```
#include <stdio.h>  
  
int main()  
{  
    printf("hello world\n");  
    return 0;  
}
```

Verschiedene Sprachkonzepte

Programmiersprachen unterteilt man in s.g.:

- **Imperative Programmiersprachen** –
Beschreibung einer Berechnungsvorschrift durch einzelne Schritte und einen Steuerfluss

Beispiele: PASCAL, C, C++, Java

- **Deklarative Programmiersprachen** –
beschreiben das Problem oder das Ziel des Programms

Beispiele: Datenbankabfragesprache SQL,
Logische Programmiersprache PROLOG

Standardisierung für C-Sprachen

C

- K&R – C 1972, Kerningham-/Ritchie
- ANSI/ISO-C89 und C90
- ANSI/ISO C95
- ANSI/ISO-C99
- ISO/IEC C 11 (2011)
- ISO/IEC C18 (2018)

C++

- C++ 1985, Bjarne Stroustrup
- ANSI/ISO-C++ 98
erster Standard, basierend auf C90 mit C++ Erweiterungen
- ANSI/ISO-C++ 03 (2000)
- ISO/IEC C++11 (2011)
- ISO/IEC C++14 (2014)
- ISO/IEC C++17 (2017)
- ISO/IEC C++20 (...)

Die Großfamilie der C-Sprachen

C/C++ - ähnliche Sprachen:

- **Java** – C++-ähnliche objektorientierte Sprache, SUN 1995
- **C#** – C++-ähnlich, objektorientiert, Microsoft 2001
- **Objective-C** - objektorientiertes C, nicht kompatibel zu C++
- **C++.NET** mit Microsoft-Erweiterungen für .NET, s.g. „managed C++“
- **JavaScript** und **PHP** – Sprachen zur Verarbeitung innerhalb von Web-Anwendungen, prozedural mit objektorientierten Erweiterungen
- **C-Shell** – Skriptsprache zur UNIX-Shell-Programmierung

... und viele andere, die von C beeinflusst wurden

Programmiersprache C

Imperative Programmiersprache:

C-Anweisungen werden in der im Programm angegebenen Reihenfolge ausgeführt:

- zeilenweise (von oben nach unten)
- innerhalb Zeile möglicherweise mehrere Anweisungen, dann von links nach rechts
- Steuerfluss-Konstrukte (*if, else, for, while, do, break, continue, return*) zur Beeinflussung der Abarbeitungsreihenfolge

Zum Vergleich: manch andere Programmiersprachen (logische und funktionale, beispielsweise PROLOG) arbeiten die Ausdrücke nicht notwendigerweise in der im Programm angegebenen Reihenfolge ab.

Programmiersprache C

Ein Programm besteht aus:

Variablenvereinbarungen: hier werden Bezeichner für die Verarbeitungselemente festgelegt. Es wird ein Typ für jede Variable angegeben, z.B. Ganzzahl (int) oder Zeichen (char)

Anweisungen:

- zur Verarbeitung der Variablen
- zur Beeinflussung des Steuerflusses

Mehrfach auftretende Anweisungsfolgen werden oft in **Funktionen** gekapselt, die Unterprogrammen in C darstellen.

Eine Hauptfunktion (**main-Funktion**) ist der Punkt, an dem mit der Ausführung eines C-Programms begonnen wird.

Programmiersprache C

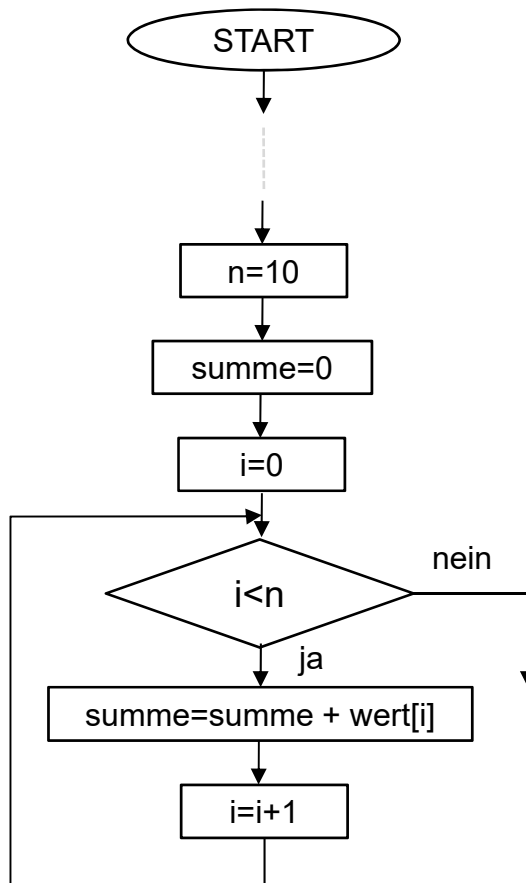
C-Programme werden in ein Binärprogramm übersetzt
Das Übersetzerprogramm ist der s.g. Compiler.

Die Übersetzung verlangt ein syntaktisch richtiges C-Programm,
d. h. das Programm muss sprachlichen Regeln genügen.

Andere Programmiersprachen werden zum Teil interpretiert,
d.h. Anweisungen werden schrittweise übersetzt und ausgeführt.

Erzeugung eines ausführbaren Programms

Algorithmus → → → C-Programm → → → x86-Maschinencode



The screenshot shows the Microsoft Visual Studio IDE with two panes. The left pane displays the C source code for a program that calculates the sum of numbers from 1 to 10. The right pane shows the corresponding x86 assembly code generated by the compiler.

C-Quelldatei.c

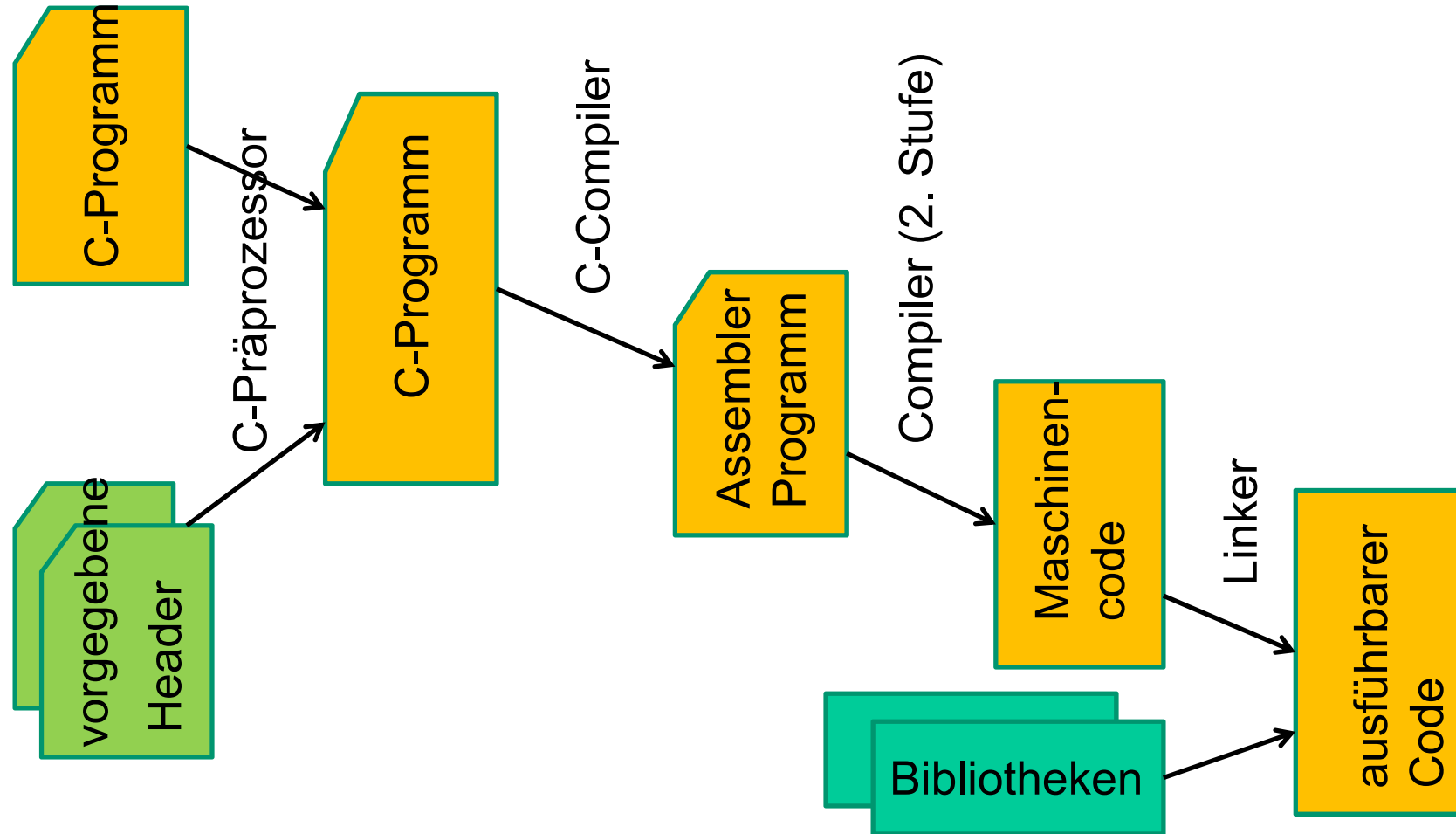
```
1 #include <stdio.h>
2
3 int main()
4 {
5     int werte[10] = {1,2,3,4,5,6,7,8,9,0};
6     int summe;
7     int i, n;
8     double mw;
9
10    n=10;
11    summe=0;
12
13    // for-Zyklus wird in cod-Datei
14    // in Zeilen 92-105 dargestellt
15    for(i=0; i<n; i=i+1)
16        summe = summe + werte[i];
17
18    // for-Zyklus wird in cod-Datei
19    // in Zeilen 92-105 dargestellt
20
21    mw = (double) summe / (double) n;
```

C-Quelldatei.cod

```
90 00 00 mov     DWORD PTR _i$[ebp], 0
91 00083 eb 09 jmp     SHORT $LN3@main
92 $LN2@main:
93 00085 8b 45 b8 mov     eax, DWORD PTR _i$[ebp]
94 00088 83 c0 01 add     eax, 1
95 0008b 89 45 b8 mov     DWORD PTR _i$[ebp], eax
96 $LN3@main:
97 0008e 8b 45 b8 mov     eax, DWORD PTR _i$[ebp]
98 00091 3b 45 ac cmp     eax, DWORD PTR _n$[ebp]
99 00094 7d 0f jge     SHORT $LN1@main
100 ; Line 13
101 00096 8b 45 b8 mov     eax, DWORD PTR _i$[ebp]
102 00099 8b 4d c4 mov     ecx, DWORD PTR _summe$[ebp]
103 0009c 03 4c 85 d0 add     ecx, DWORD PTR _werte$[ebp], ecx
104 000a0 89 4d c4 mov     DWORD PTR _summe$[ebp], ecx
105 000a3 eb e0 jmp     SHORT $LN2@main
106 $LN1@main:
107 ; Line 15
108 000a5 f2 0f 2a 45 c4 cvtsi2sd xmm0, DWORD PTR _summe$[ebp]
109 000aa f2 0f 2a 4d ac cvtsi2sd xmm1, DWORD PTR _n$[ebp]
110 000af f2 0f 5e c1 divsd  xmm0, xmm1
111 000b3 f2 0f 11 45 9c movsd  QWORD PTR _mw$[ebp], xmm0
```

Programmiersprache C

Vor dem Ausführen des Programms:



Aufbau eines C-Programms

```
// Beispielprogramm fakultaet.c  
#include <stdio.h>  
  
int main()  
{ int fakultaet;  
  int i,n;  
  
  printf("Geben Sie bitte n ein >");  
  scanf("%d",&n);  
  
  fakultaet = 1;  
  for (i=2;i<=n;i++)  
    fakultaet = fakultaet * i;  
  
  printf("Die Fakultaet von %d betraegt %d \n",  
        n, fakultaet);  
  return 0;  
}
```

The diagram illustrates the structure of a C program with the following labels and their corresponding code elements:

- Kommentar**: Points to the first line of the program: `// Beispielprogramm fakultaet.c`
- Include-Präprozessor-Anweisung**: Points to the second line: `#include <stdio.h>`
- main-Funktion**: Points to the opening brace of the `main` function: `{`
- Variablen-Deklarationen**: Points to the variable declarations: `int fakultaet;` and `int i,n;`
- Anweisungen**: Points to the assignment statement: `fakultaet = 1;`
- Steuerfluss-Konstrukte**: Points to the `for` loop and the `printf` statement: `for (i=2;i<=n;i++)` and `printf("Die Fakultaet von %d betraegt %d \n", n, fakultaet);`

C-Programm als Text

Ein C-Programm wird in einer (oder mehreren) Quelldatei(en) als Text niedergeschrieben.

Die Sprache C nutzt ein Alphabet

- aus Buchstaben a-z, A-Z,
- aus Ziffern 0-9
- und aus Sonderzeichen, z.B. !,?, (,), _

Aus dem Alphabet werden Bezeichner, Schlüsselworte, Konstanten, Zeichenketten, Operatoren, Trennzeichen und Kommentare gebildet.

Groß und Kleinschreibung wird unterschieden.

Kommentare

Kommentare in einem C-Programm werden beim Übersetzen übergangen. Sie haben keine Auswirkung auf das Programm.

Kommentare sind aber nützlich, um Erklärungen im Programm zu notieren. Oft dienen Kommentare der Dokumentation von Programmteilen.

Beginn und Ende durch */** und **/*

/ Das ist ein Kommentar */*

/ Das ist ein mehr-
zeiliger Kommentar */*

// Das ist ein einzeiliger Kommentar

Kommentare

```
/* ----- */
/* Programm zur Berechnung eines Zylindervolumens */
/* d: Durchmesser Grundkreis, h: Hoehe */
/* ----- */
#include <stdio.h>
int main()
{
    double d, h, gf, v;
    printf("d:");
    scanf("%Lf",&d);
    printf("h:");
    scanf("%Lf",&h);

    // Grundflaeche
    gf = 3.14159 * d*d / 4;
    // Volumen
    v= gf*h;
    printf("Grundflaeche: %Lf, Volumen: %Lf \n", gf, v);
    return 0;
}
```

Formatfreiheit

C ist eine formatfreie Sprache, d.h. das Aussehen, die Zeilenumbrüche oder Einrückungen haben keinen Einfluss auf die Funktion.

Beispiele:

```
int main()
{
    int a;
    scanf("%d",&a);
    printf(" quadrat = %d\n", a*a);
    return 0;
}
```

```
int main() { int a; scanf("%d",&a);
printf(" quadrat = %d\n", a*a);
return 0; }
```

Trotzdem sollte das Programm wie links dargestellt aussehen, um bessere Lesbarkeit zu gewährleisten.