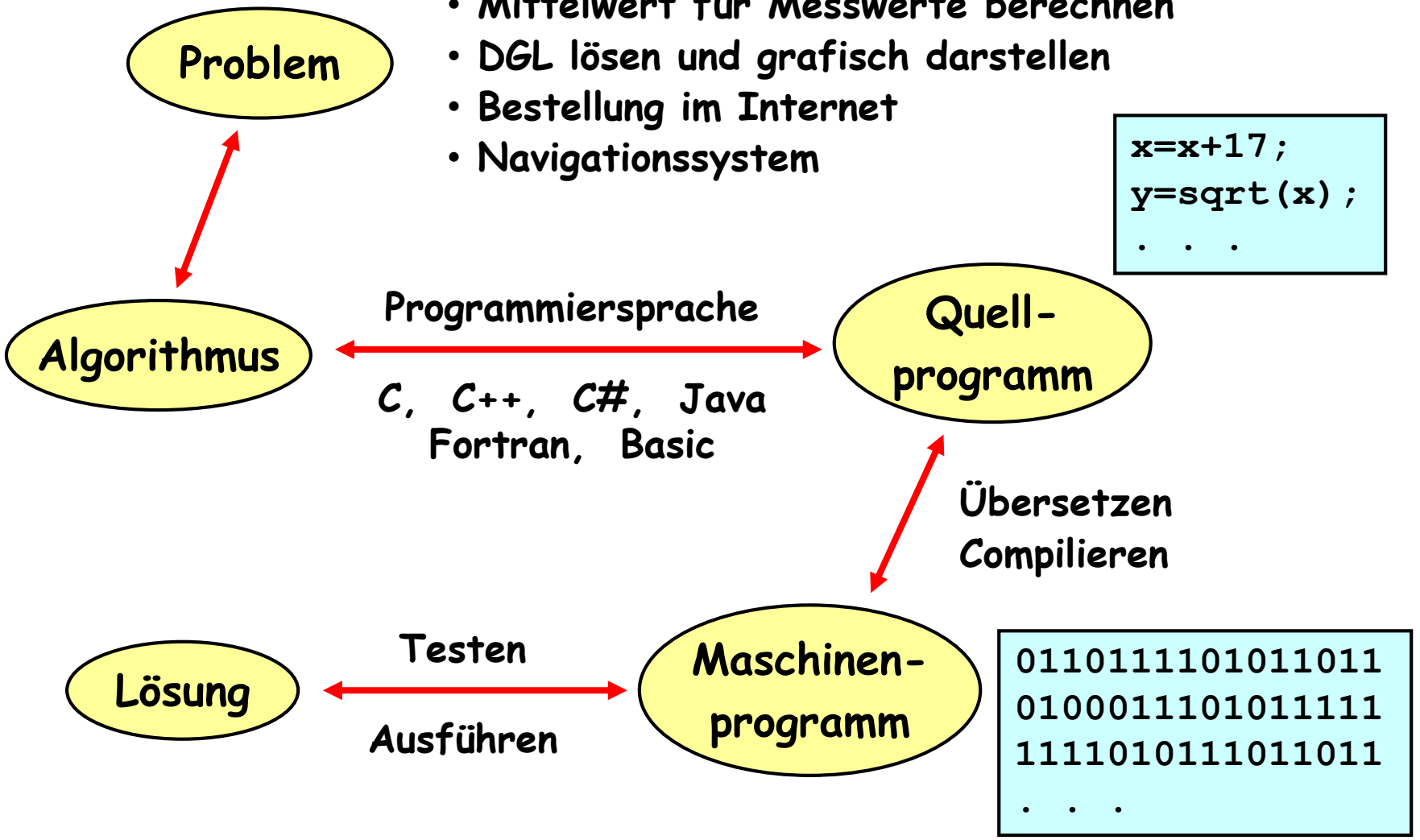
Kapitel 3

„Mein erstes C-Programm“

Kapitel 3 – „Mein erstes C-Programm“

3.1 Einleitung
3.2 Mein erstes C-Programm
3.3 Zusammenfassung

3. „Mein erstes C-Programm“



Algorithmus, Lösungsweg:

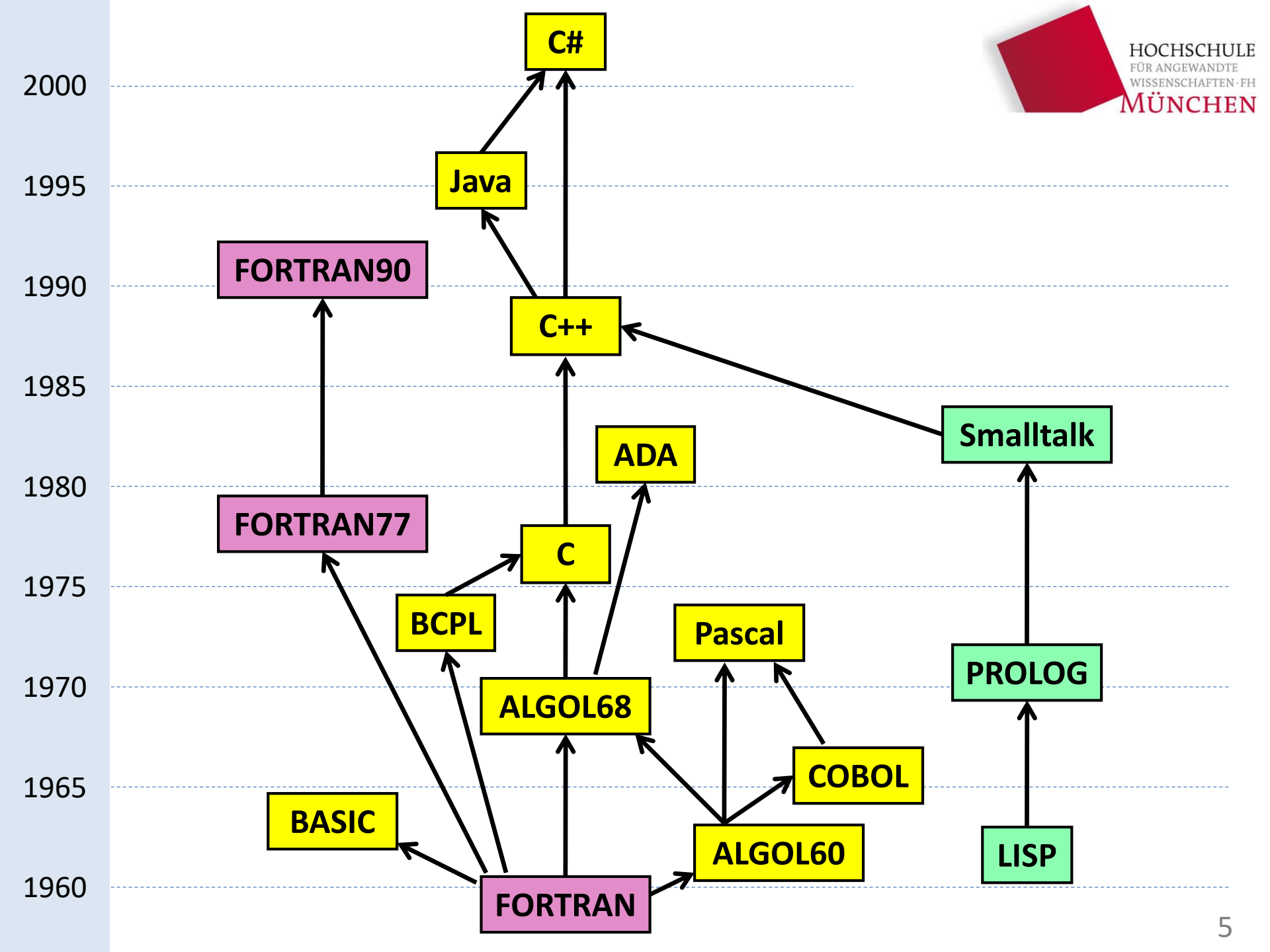
- Methode (Kochrezept) zur Lösung eines Problems, die **eindeutig** ist, aus **elementar ausführbaren Schritten** besteht und die **endlich** ist.
- Eine Verarbeitungsvorschrift, die so präzise formuliert ist, dass sie von einem mechanisch oder elektronisch arbeitenden Gerät ausgeführt werden kann. (Informatik-Duden)

Programm:

- Eine Folge von Anweisungen, durch die die Verarbeitung von Daten in einem Computer gesteuert wird.
- Formulierung eines Algorithmus und der zugehörigen Daten.

Programmiersprache:

- Eine Sprache zur Formulierung von Algorithmen und Datenstrukturen (d. h. Programmen) in einer für den Computer geeigneten Form (Schnitt-stelle zwischen Benutzer und Computer).
- Es müssen **Syntax** und **Semantik** einer Programmiersprache eindeutig definiert sein, damit man prüfen kann, welche Zeichenfolgen als Programm zugelassen sind (Syntax) und was sie auf dem Rechner bewirken (Semantik).


2000
1995
1990
1985
1980
1975
1970
1965
1960
C#
Java
FORTRAN90
C++
Smalltalk
ADA
FORTRAN77
C
BCPL
Pascal
PROLOG
ALGOL68
COBOL
BASIC
ALGOL60
LISP
FORTRAN

3.1. Einleitung

Dennis Ritchie (standing) and Ken Thompson begin porting UNIX to the PDP-11

Dennis Ritchie
"Erfinder" von C

(* 9. September 1941 in Bronxville, New York; † am 12. Oktober 2011)

Warum Programmierung in C…?

C wird in der Praxis sehr häufig verwendet:

- Programmierung von Steuergeräten (Embedded Systems)
  zum Beispiel in Fahrzeugen, Handys, Robotern
- Systemprogrammierung

C ermöglicht das Erstellen sehr effizienter Programme:

- Wenig Speicherplatzverbrauch
- Schnell – viele Anwendungen in Steuergeräten sind zeitkritisch

C unterstützt eine Vielzahl von Mikroprozessoren und Steuergeräten:

- Es gibt C-Compiler für die entsprechende Hardware und eine Vielzahl von Hilfsmittel, um die Programmierung zu unterstützen, zum Beispiel:
- Programmbibliotheken (fertige Programme für DGL, Matrizen, Statistik, FEM)
- Codegenerierung mit MATLAB für verschiedenste Steuergeräte

Betriebssysteme werden häufig in C programmiert, zum Beispiel Linux.

Viele Elemente von C finden sich auch in anderen Programmiersprachen, zum Beispiel in C++, Java, Visual Basic und C#.

Kapitel 3 – „Mein erstes C-Programm“

3.1 Einleitung
3.2 Mein erstes C-Programm
3.3 Zusammenfassung

Aufgabe:

Schreibe ein C-Programm, das die zwei Zahlen 3 und 4 addiert und das das Ergebnis am Bildschirm ausgibt.

Ziel:

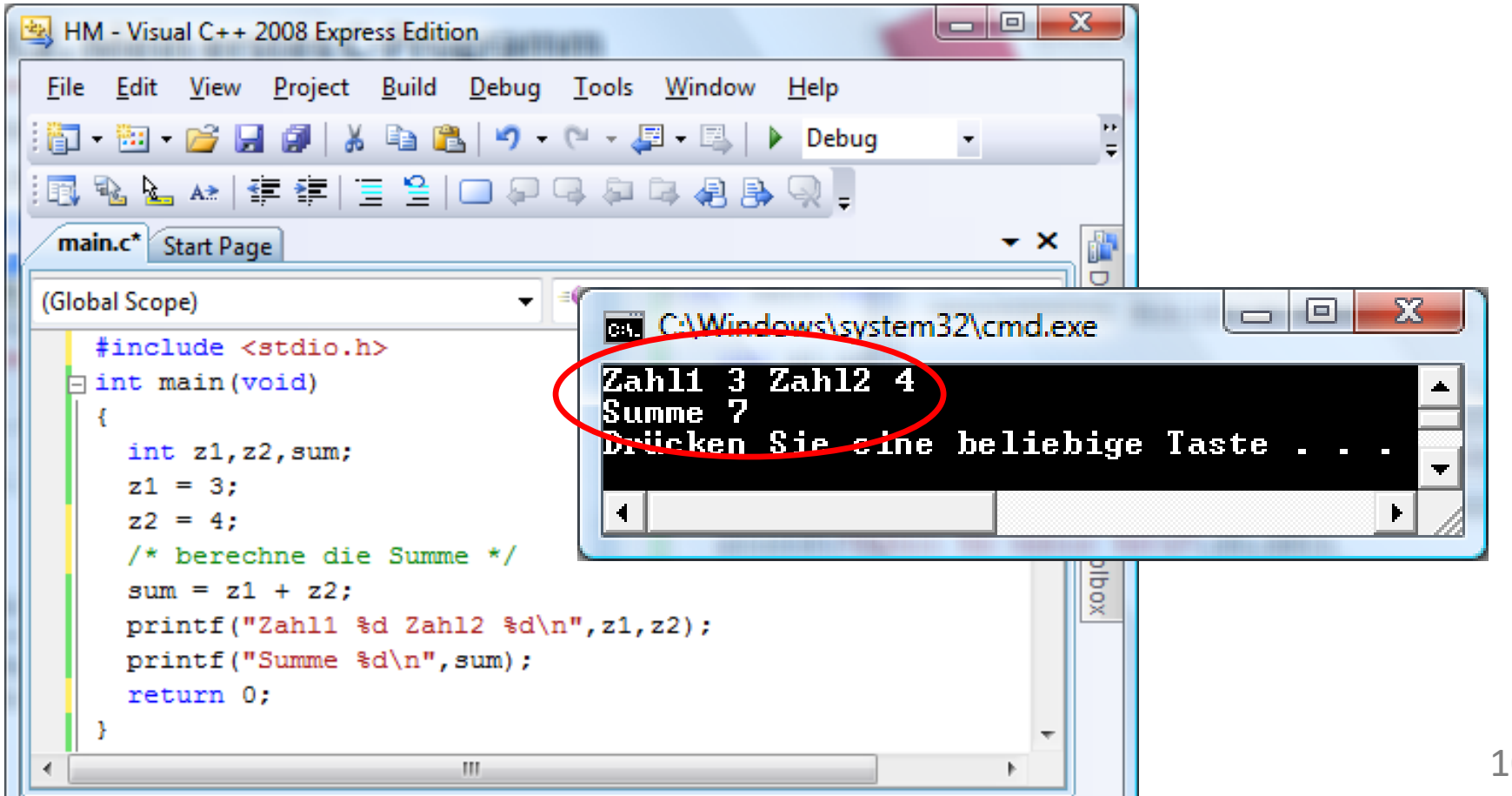
- Welche Schritte müssen am Rechner durchgeführt werden, um ein Programm zu erstellen, zu übersetzen und auszuführen?
- Wie sieht das Programm für dieses Problem aus?
- Wie sieht ein typisches C-Programm aus?
- Danach soll das Programm noch verbessert werden.

3.2. Mein erstes C-Programm

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

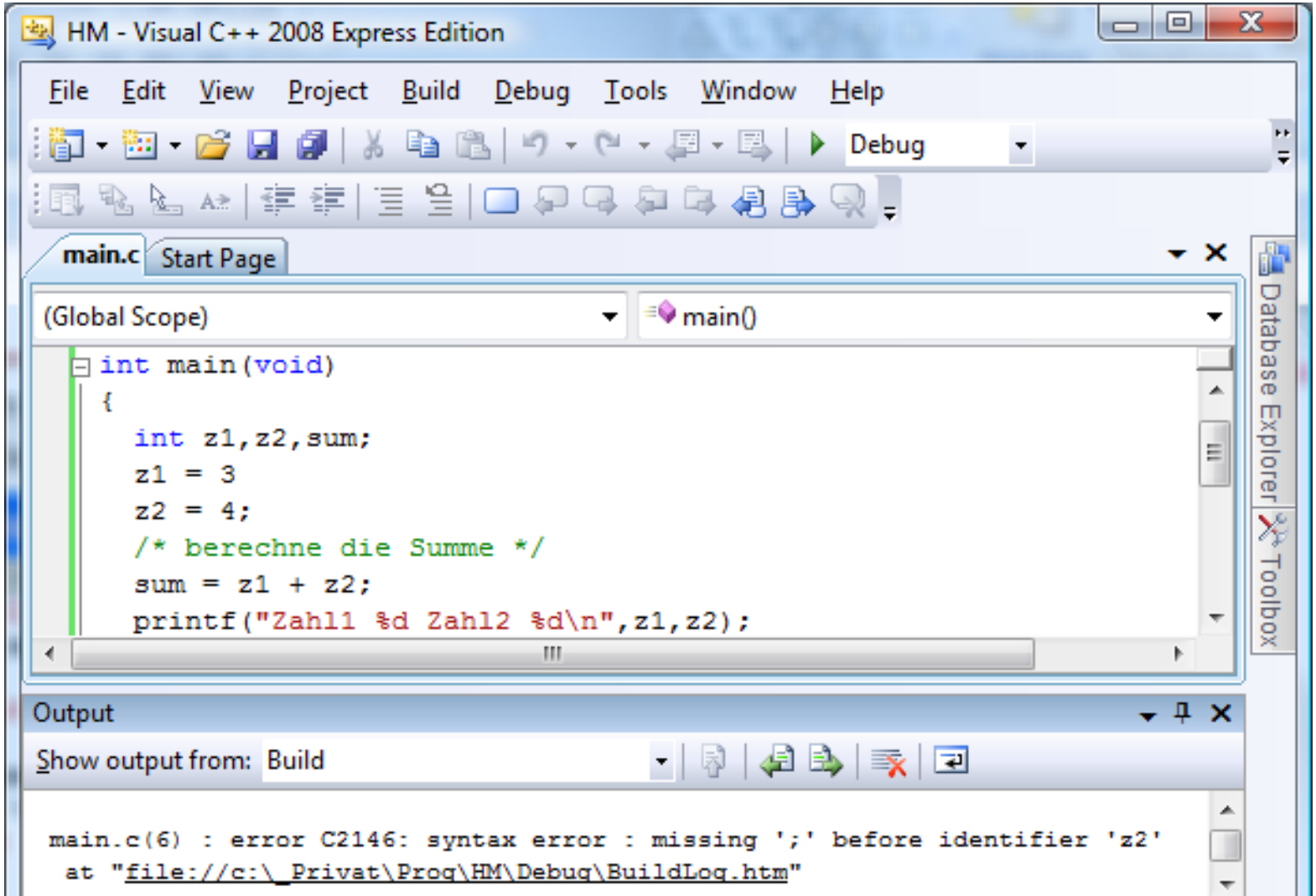
Es müssen 3 Schritte durchgeführt werden:

1. C-Quelltext mittels Editor eingeben und speichern (**`addition.c`**).
2. Das Programm in Maschinesprache übersetzen (**`addition.exe`**).
3. Jetzt kann das Programm gestartet werden.



3.2. Mein erstes C-Programm

Eine **Programmierumgebung** ist ein Hilfsmittel, um Programme zu erstellen, zu verwalten, zu testen und auszuführen.



Bekannte Programmierumgebungen:

Qt Creator:

- http://www.qt.io
- Kostenloser Download im Internet (Lizenz: GPL)
- Für Windows, Mac OS X und Linux verfügbar
- Wird im Praktikum verwendet, zu Hause installieren!

Microsoft Visual Studio:

- Kostenlose Community-Edition
- Für Microsoft Windows verfügbar
- https://www.visualstudio.com/de/vs/community/

C-Programme, die mit einer dieser Programmierumgebungen erstellt werden, laufen auch unter anderen Umgebungen, wenn man sich an die im ISO/IEC 9899-Standard beschriebenen Regeln hält. In speziellen Umgebungen, z. B. Programmierung von Steuergeräten, stehen nicht alle Funktionen zur Verfügung. Beispiel: Bildschirm-Ausgabebefehle…

3.2. Mein erstes C-Programm

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

```
#include <stdio.h>

int main(void)

{

  int z1,z2,sum;

  z1 = 3;

  z2 = 4;

  /* berechne die Summe */

  sum = z1 + z2;

  printf("Zahl1 %d Zahl2 %d\n",z1,z2);

  printf("Summe %d\n",sum);

  return 0;

}
```

Variablendefinition

z1, z2 und sum heißen Variablen

- **Speicherplatz reservieren**
- **Datentyp festlegen (int - Integer)**

Wertzuweisung

Wert in Speicherzelle schreiben

Arithmetischer Ausdruck

Wert der rechten Seite berechnen und dann der linken Seite zuweisen

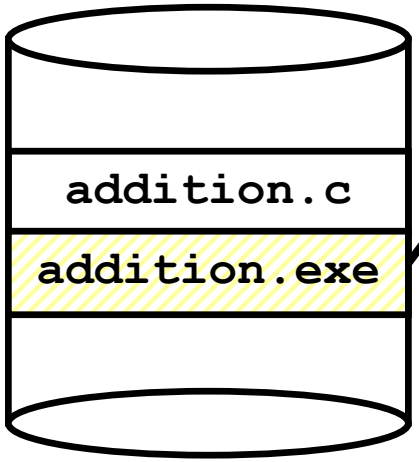
printf : Ausgabe am Bildschirm
Inhalt von " . . . " ausgeben
%d : Formatelement - dezimale Ausgabe
\n : Steuerzeichen - neue Zeile beginnen

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

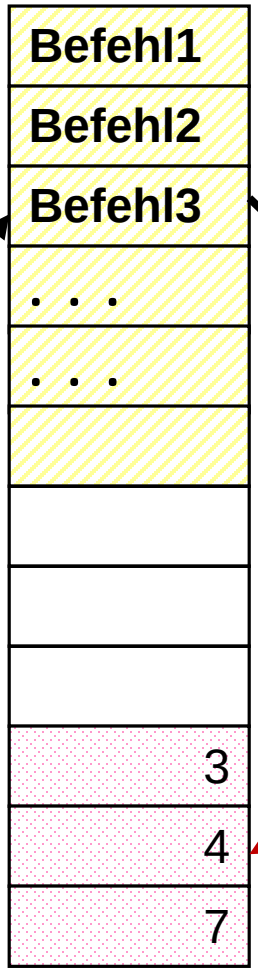
Ablauf im Rechner:

1. Programm in den Hauptspeicher laden

2. Maschinenbefehle nacheinander abarbeiten

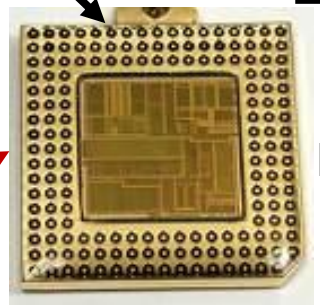


Festplatte



Hauptspeicher

Ausgabe



Prozessor

Variablen lesen und verändern

3.2. Mein erstes C-Programm

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

```
#include <stdio.h>    /* k2addit1.c */
int main (void)
{
  int z1, z2, summe;
  printf("Erste Zahl eingeben:\n");
  scanf("%d",&z1);
  printf("Zweite Zahl eingeben:\n");
  scanf("%d",&z2);
  /* berechne die Summe */
  summe = z1 + z2;
  printf("Zahl1 %d Zahl2 %d\n",z1,z2);
  printf("Summe : %d \n",summe);
  return 0;
}
```

scanf

das Programm wartet solange, bis eine Eingabe erfolgt ist, interpretiert die Eingabe als ganze Zahl (%d) und speichert das Ergebnis in der Speicher-zelle für z1 (&z1) ab.

3.2. Mein erstes C-Programm

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

```
#include <stdio.h>   /* k2addit2.c */
int main(void)
{
  float z1, z2, sum;
  printf("Erste Zahl eingeben :\n");
  scanf("%f", &z1);
  printf("Zweite Zahl eingeben :\n");
  scanf("%f", &z2);
  sum = z1 + z2;
  printf("Zahl1:%f Zahl2:%f\n",z1,z2);
  if(sum < 0.0)
  {
    printf("Ergebnis ist negativ!");
  }
  else
  {
    printf("Summe : %f \n",sum);
    printf("Ergebnis ist positiv!");
  }
  return 0;
}
```

float
Speicherplatz für eine Fließ-kommazahl reservieren

%f
Formatelement für eine Fließkommanzahl

sum < 0
Bedingung – entweder wahr (erfüllt) oder falsch

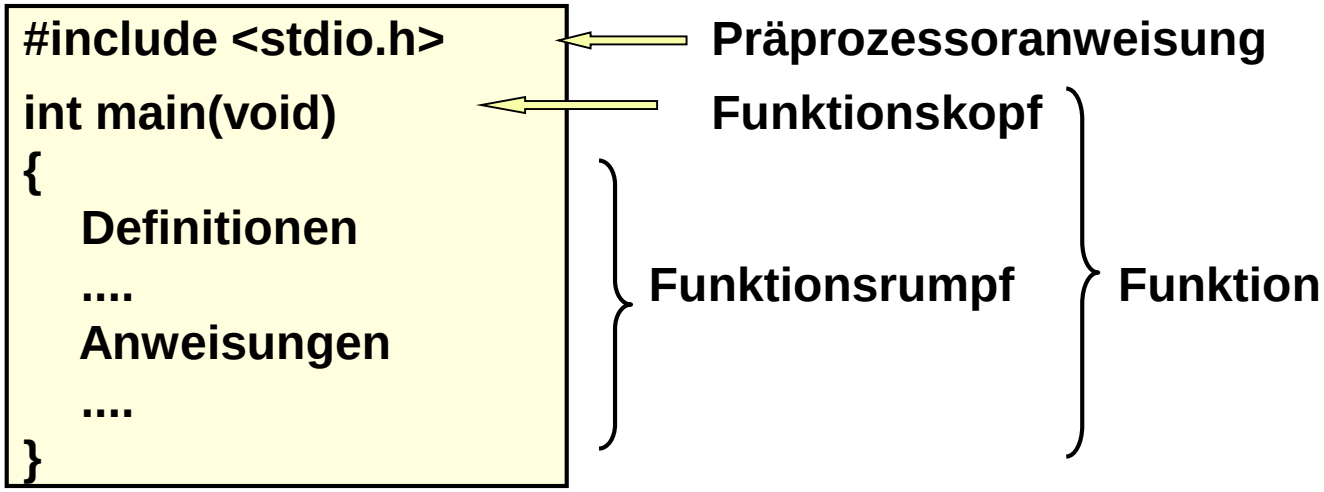
if - else - Kontrollstruktur
wenn die Bedingung erfüllt ist, dann nur den ersten Teil ausführen ansonsten nur den zweiten Teil (Alternative)

Kapitel 3 – „Mein erstes C-Programm“

3.1 Einleitung
3.2 Mein erstes C-Programm
3.3 Zusammenfassung

3.3. Zusammenfassung

a) Struktur eines einfachen C-Programms



Ein C-Programm besteht mindestens aus der Funktion **main** (Hauptprogramm).

Eine **Funktion** besteht aus:

- Funktionskopf – **main** ist der Name der Funktion
- Funktionsrumpf

Der **Funktionsrumpf** beginnt mit **{** und endet mit **}** und enthält:

- Definition von Variablen
- Anweisungen – zum Beispiel arithmetische Ausdrücke, Wertzuweisungen
- Kommentare – beginnen mit **/*** und enden mit ***/**

b) Definition von Variablen

int z; Ganze Zahl mit Vorzeichen
float x, y; Fließkommazahl

Die Anweisung

int z;

definiert eine Variable vom Typ Integer (ganze Zahl mit Vorzeichen).

- Bei der Variablendefinition wird ein Datentyp wird festgelegt, hier: Integer
- Speicherplatz wird reserviert, um den Wert einer ganzen Zahl speichern zu können (für eine Integer-Variable typischerweise 4 Bytes)
- z kann nur ganze Zahlen speichern, keine Gleitkommazahlen
- z ist der Name der Variablen
- Der Variablenname ist frei wählbar, muss aber eindeutig sein
- Inhalt der Speicherzelle (Wert) ist unter dem Namen z ansprechbar

c) Anweisungen

```
z = 17;
sum = z1 + z2;
```

Beide Anweisungen sind **Wertzuweisungen**. Der Operator = ist der **Zuweisungsoperator** (= ist kein mathematisches Gleichheitszeichen !!!).

Bedeutung von =

1. Berechne zuerst den Wert der rechten Seite
2. Weise danach das Ergebnis der linken Seite zu

Beispiel:

```
z = z + 5;
```

Erklärung:

1. Hole den aktuellen Wert aus der Speicherzelle z („hole den aktuellen Wert der Variablen z“)
2. Addiere zu diesem Wert 5 hinzu
3. Speichere danach das Ergebnis in der Speicherzelle z

Der alte Wert von z wird überschrieben und ist damit verloren.

3.3. Zusammenfassung

d) Operatoren

+ - * / %	Arithmetische Operatoren
< >	Relationale Operatoren
=	Zuweisungsoperator
&	Adressoperator

e) Schlüsselwörter – reservierte Wörter

int , **float** , **if**, **else** sind Schlüsselwörter der Sprache C. Schlüsselwörter haben eine spezielle Bedeutung in einer Programmiersprache und dürfen nicht als Namen für Variablen oder Funktionen verwendet werden.

f) Kontrollstruktur

```
if ( Bedingung )
{
   /* Anweisungen, falls Bedingung wahr */
}
else
{
   /* Anweisungen, falls Bedingung falsch */
}
```

3.3. Zusammenfassung

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

g) Ein- und Ausgabe

Ausgabe am Bildschirm mit printf:

printf("..."); Text " ... " am Bildschirm ausgeben

printf("...%d...", z); Text " ... " und aktuellen Wert von z ausgeben

Einlesen von der Tastatur mit scanf:

scanf("%d", &z);

Formatierungszeichen („Platzhalter“ für Variablen):

%d : Ein-/Ausgabe ganzer Zahlen in der Dezimaldarstellung

%f : Ein-/Ausgabe von Gleitkommazahlen

Steuerzeichen:

\n : Zeilenumbruch („beginne eine neue Zeile“)

h) Sonstiges

- C unterscheidet zwischen **Groß- und Kleinschreibung**
  zahl1 ≠ **ZAHL1** ≠ **Zahl1** (drei verschiedene Namen!)
- Semikolon **;** kennzeichnet das Ende von Definitionen und Anweisungen (eine neue Zeile alleine bedeutet nicht das Ende einer Anweisung!)

3.3. Zusammenfassung

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

```
#include  < stdio.h   >
int main ( void   )
{
  int z1,
      z2, sum          ;
 z1 = 3 ;   z2 = 4 ;
        /* berechne die Summe
                          */
 sum =   z1
             + z2        ;
printf("Zahl1:%d Zahl2:%d\n",
             z1,
    z2  )    ;
     printf(
  "Summe: %d\n"     , sum) ;
  return    0   ;
      }
```

```
#include <stdio.h>
int main(void)
{
 int z1,z2,sum;
 z1 = 3;
 z2 = 4;
 /* berechne die Summe */
 sum = z1 + z2;
 printf("Zahl1:%d Zahl2:%d\n",
        z1,z2);
 printf("Summe: %d\n",sum);
 return 0;
}
```