Hochschule Bochum
Bochum University
of Applied Sciences

Angewandte Informatik

Prof. Dr. Peter Gerwinski

22. November 2012

2 Einführung in C

Sprachelemente weitgehend komplett

Es fehlen:

- Ergänzungen (z. B. ternärer Operator, **union**, **unsigned**, **volatile**)
- Bibliotheksfunktionen (z. B. malloc())

⟶ werden eingeführt, wenn wir sie brauchen

- Konzepte (z. B. rekursive Datenstrukturen, Klassen selbst bauen)

⟶ werden eingeführt, wenn wir sie brauchen, oder:

⟶ Literatur

(z. B. Wikibooks: C-Programmierung,
Dokumentation zu Compiler und Bibliotheken)

- Praxiserfahrung

⟶ Praktikum: nur Einstieg

⟶ selbständig arbeiten

3 Bibliotheken

3.1 Der Präprozessor

#include: Text einbinden

- **#include** <stdio.h>: Standard-Verzeichnisse – Standard-Header
- **#include** "answer.h": auch aktuelles Verzeichnis – eigene Header

3 Bibliotheken

3.1 Der Präprozessor

#include: Text einbinden

- **#include** <stdio.h>: Standard-Verzeichnisse – Standard-Header
- **#include** "answer.h": auch aktuelles Verzeichnis – eigene Header

#define VIER 4: Text ersetzen lassen – Konstante definieren

3 Bibliotheken

3.1 Der Präprozessor

#include: Text einbinden

- **#include** <stdio.h>: Standard-Verzeichnisse – Standard-Header
- **#include** "answer.h": auch aktuelles Verzeichnis – eigene Header

#define VIER 4: Text ersetzen lassen – Konstante definieren

- Kein Semikolon!

3 Bibliotheken

3.1 Der Präprozessor

#include: Text einbinden

- **#include** <stdio.h>: Standard-Verzeichnisse – Standard-Header
- **#include** "answer.h": auch aktuelles Verzeichnis – eigene Header

#define VIER 4: Text ersetzen lassen – Konstante definieren

- Kein Semikolon!
- Berechnungen in Klammern setzen:
  #define VIER (2 + 2)

3 Bibliotheken

3.1 Der Präprozessor

#include: Text einbinden

- **#include** <stdio.h>: Standard-Verzeichnisse – Standard-Header
- **#include** "answer.h": auch aktuelles Verzeichnis – eigene Header

#define VIER 4: Text ersetzen lassen – Konstante definieren

- Kein Semikolon!
- Berechnungen in Klammern setzen:
  #define VIER (2 + 2)
- Konvention: Großbuchstaben

3 Bibliotheken

3.2 Bibliotheken einbinden

Inhalt der Header-Datei: externe Deklarationen

```
extern int answer (void);
extern int printf (__const char *__restrict __format, ...);
```

Funktion wird „anderswo“ definiert

3 Bibliotheken

3.2 Bibliotheken einbinden

Inhalt der Header-Datei: externe Deklarationen

```
extern int answer (void);
extern int printf (__const char *__restrict __format, ...);
```

Funktion wird „anderswo“ definiert

- separater C-Quelltext: mit an `gcc` übergeben
- vorcompilierte Bibliothek: `-lfoo`
  = Datei libfoo.a in Standard-Verzeichnis

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

```
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

```
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
```

- Compiler-Aufruf:

```
gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

  ```
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
  ```

- Compiler-Aufruf:

  ```
  gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
  ```

- Funktionen aufrufen: glutInit (&argc, argv);

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

  ```
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
  ```

- Compiler-Aufruf:

  ```
  gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
  ```

- Funktionen aufrufen: glutInit (&argc, argv);
- Konstante: GLUT_RGBA

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

  ```
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
  ```

- Compiler-Aufruf:

  ```
  gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
  ```

- Funktionen aufrufen: glutInit (&argc, argv);
- Konstante: GLUT_RGBA
- Datentypen: GLfloat

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

  ```
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
  ```

- Compiler-Aufruf:

  ```
  gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
  ```

- Funktionen aufrufen: glutInit (&argc, argv);
- Konstante: GLUT_RGBA
- Datentypen: GLfloat
- Array übergeben:

  ```
  GLfloat light0_position[] = {1.0, 2.0, −2.0, 1.0};
  glLightfv (GL_LIGHT0, GL_POSITION, light0_position);
  ```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

- Include-Dateien:

  ```
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
  ```

- Compiler-Aufruf:

  ```
  gcc -Wall -O cube.c -lGL -lGLU -lglut -o cube
  ```

- Funktionen aufrufen: glutInit (&argc, argv);
- Konstante: GLUT_RGBA
- Datentypen: GLfloat
- Array übergeben:

  ```
  GLfloat light0_position[] = {1.0, 2.0, −2.0, 1.0};
  glLightfv (GL_LIGHT0, GL_POSITION, light0_position);
  ```

- Funktion übergeben (Callbacks):

  ```
  void draw_cube (void)
  { ... }

  glutDisplayFunc (draw_cube);
  ```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

```
glutSolidCube (GLdouble size);
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

```
glutSolidCube (GLdouble size);
glutWireCube (GLdouble size);
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

glutSolidCube (GLdouble size);
glutWireCube (GLdouble size);
glutSolidSphere (GLdouble radius, GLint slices, GLint stacks); ... Wire ...

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

```
glutSolidCube (GLdouble size);
glutWireCube (GLdouble size);
glutSolidSphere (GLdouble radius, GLint slices, GLint stacks); ... Wire ...
glutSolidCone (GLdouble base, GLdouble height, GLint slices, GLint stacks);
glutSolidTorus (GLdouble innerRadius, GLdouble outerRadius,
                GLint sides, GLint rings);
glutSolidDodecahedron (void);
glutSolidOctahedron (void);
glutSolidTetrahedron (void);
glutSolidIcosahedron (void);
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

```
glutSolidCube (GLdouble size);
glutWireCube (GLdouble size);
glutSolidSphere (GLdouble radius, GLint slices, GLint stacks); ... Wire ...
glutSolidCone (GLdouble base, GLdouble height, GLint slices, GLint stacks);
glutSolidTorus (GLdouble innerRadius, GLdouble outerRadius,
                GLint sides, GLint rings);
glutSolidDodecahedron (void);
glutSolidOctahedron (void);
glutSolidTetrahedron (void);
glutSolidIcosahedron (void);
glutSolidTeapot (GLdouble size);
```

3.3 Bibliothek verwenden (Beispiel: OpenGL)

Aufgabe:

Für welche elementaren geometrischen Körper
stellt die GLUT-Bibliothek Zeichenroutinen zur Verfügung?

Lösung:

```
glutSolidCube (GLdouble size);
glutWireCube (GLdouble size);
glutSolidSphere (GLdouble radius, GLint slices, GLint stacks); ... Wire ...
glutSolidCone (GLdouble base, GLdouble height, GLint slices, GLint stacks);
glutSolidTorus (GLdouble innerRadius, GLdouble outerRadius,
                GLint sides, GLint rings);
glutSolidDodecahedron (void);
glutSolidOctahedron (void);
glutSolidTetrahedron (void);
glutSolidIcosahedron (void);
glutSolidTeapot (GLdouble size);
glutSolidRhombicDodecahedron (void);
glutSolidSierpinskiSponge (int num_levels, GLdouble offset[3], GLdouble scale);
glutSolidCylinder (GLdouble radius, GLdouble height, GLint slices, GLint stacks);
```

Praktikumsaufgabe

Schreiben Sie ein 3d-„Basketball“-Programm.

Ein schief abgeworfener Ball fliegt
entlang einer Wurfparabel
durch einen horizontalen Ring.