

Debugging

C - Kurs 2010

Alexander Eichner

www.freitagrunde.org

14. September 2010



This work is licensed under the *Creative Commons Attribution-NonCommercial-ShareAlike 3.0 License*.

9/9

0800 Antam started
 1000 " stopped - antam ✓
 1300 (032) MP - MC 1.98217000
 (033) PRO 2 2.130476415
 convd 2.130676415

{ 1.2700 9.037 847 025
 9.037 846 995 convd
 4.615925059(-2)

Relays 6-2 in 033 failed special speed test
 in relay 10,000 test.

Relay 3145
 Relay 3376

1100 Started Cosine Tape (Sine check)
 1525 Started Multi-Adder Test.

1545 Relay #70 Panel F
 (moth) in relay.



First actual case of bug being found.
 1630 Antam started.
 1700 closed down.

Abbildung: Erster gefundener Bug (1947).

Quelle: <http://de.wikipedia.org/wiki/Datei:H96566k.jpg>

1 Wiederholung

2 Ein wenig Theorie

3 Praxis



4!

1 Wiederholung

2 Ein wenig Theorie

3 Praxis

4!

- Theoretisch alles was man zum Schreiben eines Programms in C wissen muss

4!

- Theoretisch alles was man zum Schreiben eines Programms in C wissen muss
- Es fehlt Übung und Erfahrung, um gewisse Fehler zu vermeiden

4!

- Die Fallstricke von C kennenlernen
- Möglichkeiten des Debugging kennenlernen
- Bugs identifizieren
- einen Debugger effektiv einsetzen zu können
- euch das Leben einfacher zu machen ;)



4!

1 Wiederholung

2 Ein wenig Theorie

3 Praxis

4!

Welche Arten von Fehlern gibt es?

- Programm kompiliert gar nicht erst - Syntaxfehler



4!

Welche Arten von Fehlern gibt es?

- Programm kompiliert gar nicht erst - Syntaxfehler
- Es stürzt ab (SIGSEGV oder SIGBUS) - Laufzeitfehler



4!

Welche Arten von Fehlern gibt es?

- Programm kompiliert gar nicht erst - Syntaxfehler
- Es stürzt ab (SIGSEGV oder SIGBUS) - Laufzeitfehler
- Es reagiert nicht mehr stürzt aber nicht ab - Laufzeitfehler



4!

Welche Arten von Fehlern gibt es?

- Programm kompiliert gar nicht erst - Syntaxfehler
- Es stürzt ab (SIGSEGV oder SIGBUS) - Laufzeitfehler
- Es reagiert nicht mehr stürzt aber nicht ab - Laufzeitfehler
- Es macht nicht das was es soll - Logikfehler oder Designfehler

4!

Heap/Stack Overflow



4!

Heap/Stack Overflow - Programm stürzt ab

- bedeutet das das Programm auf nicht reservierten Speicher zugreift

Beispiel

```
1 char *pStr = 0xdeadbeef;  
2 printf("%s\n", pStr);
```

Ausgabe:

```
1 Segmentation fault
```

Uninitialisierte Variablen

Beispiel

```
1 int iWert;  
2 printf("wert = %d\n", iWert);
```

Ausgabe:

```
1 wert = 1337
```

Integeroverflow

Beispiel

```
1 unsigned char byte = 255;  
2 printf(" byte = %d\n", byte);  
3 byte++;  
4 printf(" byte = %d\n", byte);
```

Ausgabe:

```
1 byte = 255  
2 byte = 0
```


Integeroverflow

Beispiel

```
1 unsigned char byte = 255;  
2 printf(" byte = %d\n", byte);  
3 byte++;  
4 printf(" byte = %d\n", byte);  
5 printf("%d\n", 10/byte);
```

Ausgabe:

```
1 byte = 255  
2 byte = 0
```

Integeroverflow

Beispiel

```
1 unsigned char byte = 255;  
2 printf(" byte = %d\n", byte);  
3 byte++;  
4 printf(" byte = %d\n", byte);  
5 printf("%d\n", 10/byte);
```

Ausgabe:

```
1 byte = 255  
2 byte = 0  
3 Floating point exception
```

- Endlosschleifen

A large, stylized graphic of a gear or sun with a white circle in the center containing the text '4!'. The gear is composed of several light gray segments radiating from a central white circle. The number '4' and the exclamation mark '!' are rendered in a bold, white, sans-serif font.

- Endlosschleifen
- Speicherkorruption

A large, stylized graphic of a gear or sun with a central circle containing the number '4!'. The gear is composed of several large, light gray segments radiating from a central white circle. Inside the central circle, the number '4!' is written in a bold, white, sans-serif font.

4!

- Endlosschleifen
- Speicherkorruption - oft schwierig zu debuggen



4!

Die 4 beliebtesten Stolpersteine in C

Platz 1: = anstatt ==



4!

Die 4 beliebtesten Stolpersteine in C

Platz 1: = anstatt ==

Beispiel

```
1  int iWert = 2;  
2  if (iWert = 3)  
3      printf("iWert ist %d\n", iWert);
```

Die 4 beliebtesten Stolpersteine in C

Platz 1: = anstatt ==

Beispiel

```
1  int iWert = 2;  
2  if (iWert = 3)  
3      printf("iWert ist %d\n", iWert);
```

Ausgabe:

```
1  iWert ist 3
```


Platz 2: ; an der falschen Stelle



4!

Die 4 beliebtesten Stolpersteine in C

Platz 2: ; an der falschen Stelle

Beispiel

```
1 int iWert = 2;  
2 if (iWert == 3);  
3     printf("iWert ist 3\n");
```

Die 4 beliebtesten Stolpersteine in C

Platz 2: ; an der falschen Stelle

Beispiel

```
1 int iWert = 2;  
2 if (iWert == 3);  
3     printf("iWert ist 3\n");
```

Ausgabe:

```
1 iWert ist 3
```

Beispiel: Endlosschleife

```
1 int iCountdown = 10;
2
3 while (iCountdown > 0);
4 {
5     printf("%d\n", iCountdown);
6     iCountdown--;
7 }
```

Beispiel: Endlosschleife

```
1 int iCountdown = 10;  
2  
3 while (iCountdown > 0);  
4 {  
5     printf("%d\n", iCountdown);  
6     iCountdown--;  
7 }
```

Ausgabe:

Die 4 beliebtesten Stolpersteine in C

Platz 3: switch case ohne break



4!

Die 4 beliebtesten Stolpersteine in C

Platz 3: switch case ohne break

Beispiel

```
1  int iValue = 2;
2
3  switch (iValue)
4  {
5      case 1: printf(" iValue == 1\n" );
6      case 2: printf(" iValue == 2\n" );
7      case 3: printf(" iValue == 3\n" );
8      default:
9          printf(" iValue == %d\n" , iValue );
10 }
```

Die 4 beliebtesten Stolpersteine in C

Platz 3: switch case ohne break

Beispiel

```
1  int iValue = 2;
2
3  switch (iValue)
4  {
5      case 1: printf(" iValue = 1\n");
6      case 2: printf(" iValue = 2\n");
7      case 3: printf(" iValue = 3\n");
8      default:
9          printf(" iValue = %d\n" , iValue);
10 }
```

Ausgabe:

```
1  iValue = 2
2  iValue = 3
3  iValue = 2
```


Platz 4: Dangling else

A large, stylized graphic of a gear or sun with a central circle containing the text '4!'. The gear has several teeth, and the central circle is a solid grey color with the text '4!' in white. The background is white with grey gear teeth.

Die 4 beliebtesten Stolpersteine in C

Platz 4: Dangling else

Beispiel: Wo gehört das else hin?

```
1 int i = 2;
2 int j = 1;
3
4 if (i == 1)
5     if (j == 1)
6         printf("Both_1\n");
7 else
8     printf("iValue=%d\n", iValue);
```

Die 4 beliebtesten Stolpersteine in C

Platz 4: Dangling else

Beispiel: Wo gehört das else hin?

```
1 int i = 2;
2 int j = 1;
3
4 if (i == 1)
5     if (j == 1)
6         printf("Both_1\n");
7 else
8     printf("iValue=%d\n", iValue);
```

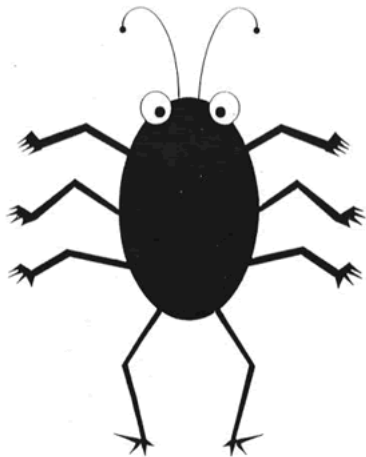
Ausgabe:

weitere findet ihr hier:

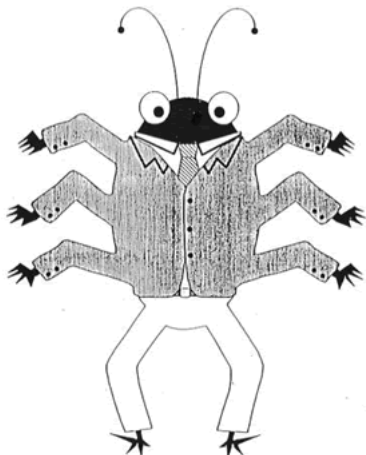
<http://literateprogramming.com/ctraps.pdf>



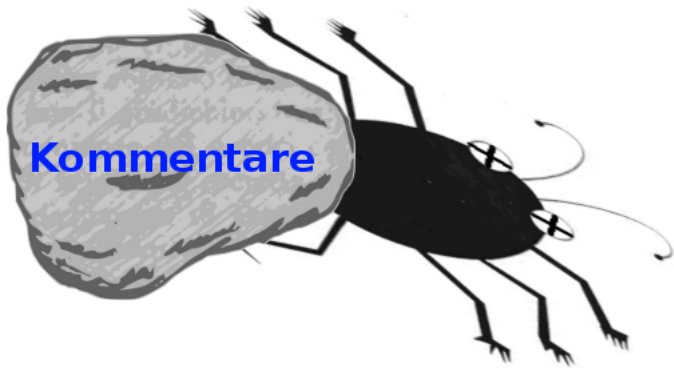
4!



BUG

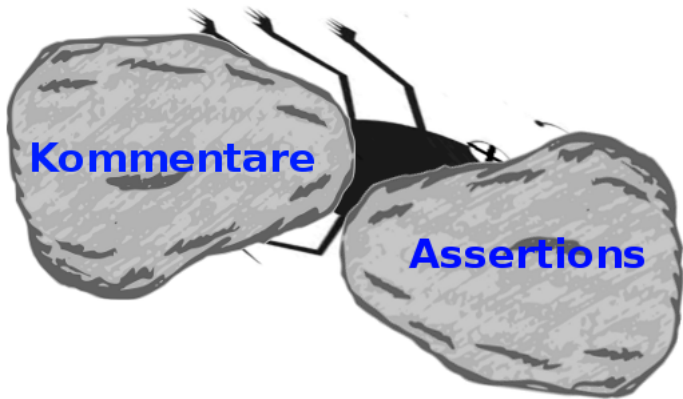


FEATURE



Kommentare - Beispiel

```
1 /**
2  * Millisecond granular sleep function.
3  *
4  * @returns IPRT status code.
5  * @retval VINF_SUCCESS on success.
6  * @retval VERR_INTERRUPTED if a signal or other
7  *       asynchronous stuff happend
8  *       which interrupt the peaceful sleep.
9  * @param  cMillies      Number of milliseconds to sleep.
10 *                0 milliseconds means yielding the
11 *                timeslice – deprecated!
12 * @remark  See RTThreadNanoSleep() for sleeping for
13 *          smaller periods of time.
14 */
15 int RTThreadSleep(RTMSINTERVAL cMillies);
```



Assertions - Beispiel

Assertions

Beispiel: assert

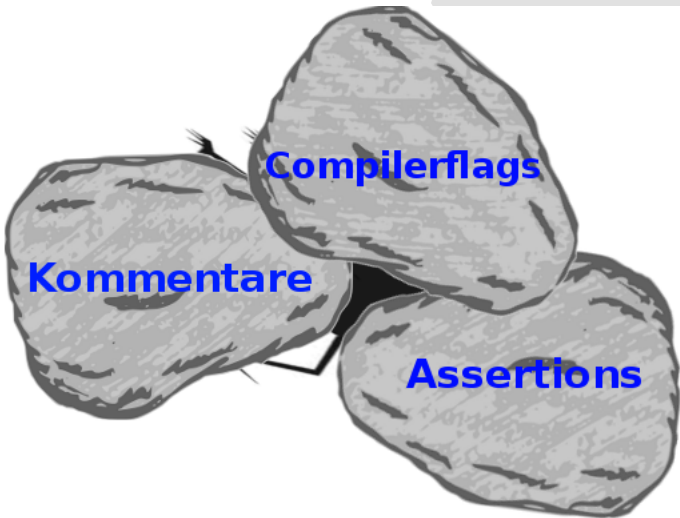
```
1 #include <assert.h>
2
3 int strlen(const char *pszStr) {
4     assert(pszStr != NULL);
5     ...
6 }
7
8 void main(void) {
9     strlen(NULL);
10 }
```

Ausgabe:

```
1 test_assert: test_assert.c:6: my_strlen: Assertion
2 pszStr != ((void *)0) failed.
3 Aborted
```

- fängt unerlaubte Zustände im Programm sehr früh ab
- -DNDEBUG deaktiviert sie im Releasecode

A large, light gray graphic in the background features a circle containing the number '4' followed by an exclamation mark '!', both in white. The circle is partially overlaid by other light gray shapes, including a large triangle and a smaller circle, creating a layered, abstract design.



Compiler flags

```
1 #include <stdio.h>
2
3 int main(int argc, char *argv[])
4 {
5     int a, b;
6
7     a = b;
8     if (a = 42);
9         printf("Answer to the Ultimate Question of Life ,
10                the Universe , and Everything!\n");
11 }
```

Ausgabe:

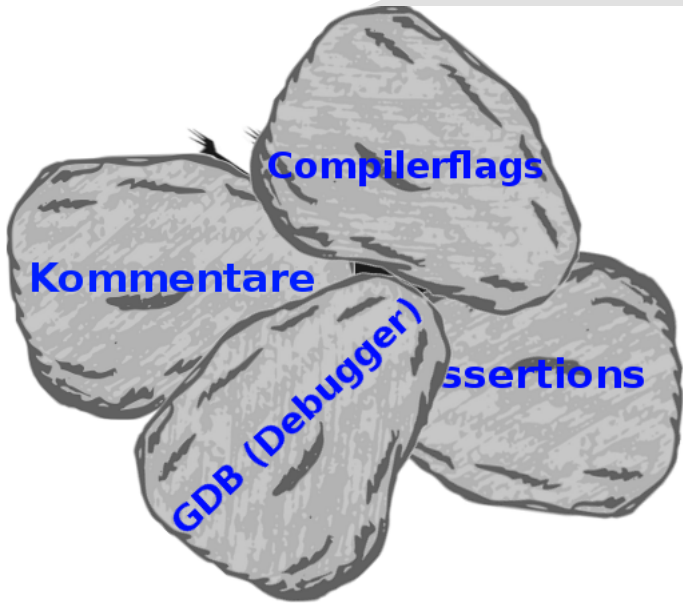
```
1 #gcc -o warnings warnings.c  
2 #
```

Ausgabe:

```
#gcc -Wall -Wextra -pedantic -o warnings warnings.c
warnings.c: In function 'main':
warnings.c:8: warning: suggest parentheses around assignment
                used as truth value
warnings.c:8: warning: suggest braces around empty body in
                an 'if' statement
warnings.c:3: warning: unused parameter 'argc'
warnings.c:3: warning: unused parameter 'argv'
warnings.c:11: warning: control reaches end of non-void
                function
warnings.c:7: warning: 'b' is used uninitialized in this
                function
```

Dokumentation zu den Optionen:

<http://gcc.gnu.org/onlinedocs/gcc/Warning-Options.html>



Was kann man mit dem gdb machen?

- erlaubt es Programme zur Laufzeit anzuhalten
- Zustand des Programmes kann untersucht und verändert werden
- noch viel mehr



4!

Vorbereitung



4!

Vorbereitung

Die richtigen Compilerflags:

```
gcc -g -O0 -o prog prog.c
```



4!

1 Wiederholung

2 Ein wenig Theorie

3 Praxis



4!

Danke

Fragen?



4!

Liste einiger gdb Kommandos

run	Ausführen eines Programms
quit	gdb beenden
kill	Programm beenden
cont	Fortfahren mit der Ausführung eines Programms
bt	Stacktrace
frame nr	Zum Frame im Stacktrace wechseln
list	Quellcode anzeigen
print	Variableninhalte anzeigen
printf	print formatted
break	Breakpoint setzen
condition	Bedingung für Breakpoint setzen
delete	Breakpoint löschen
next	Zeile im Quellcode ausführen (überspringt Funktionen)
step	Zeile im Quellcode ausführen (geht in Funktionen rein)
set print pretty on	übersichtliches Ausgeben von struct

A large, stylized graphic of a gear or sun with a central circle containing the text '4!'. The gear is composed of several curved segments, and the central circle is a solid light gray. The text '4!' is white and bold, centered within the circle.

4!