

# Debugging mit GDB

Proseminar - „C - Grundlagen und Konzepte“

22. Mai 2014 - Albrecht Oster

## Gliederung

Was ist Debugging?

Was ist GDB?

Fallbeispiel eines Problems

Wie kann uns GDB an dieser Stelle helfen?

Weitere Einsatzmöglichkeiten

2 / 10

## Was ist Debugging?

Software auf Fehler untersuchen

Testing

Produktiveinsatz

Fehlerbehebung

Kann zeitaufwendig und mühsam sein

3 / 10

## Was ist GDB?

GDB = GNU Debugger

Auf eine Vielzahl an Sprachen, Plattformen und Architekturen portiert

Kommandozeilenprogramm

In viele Entwicklungsumgebungen grafisch integriert

Erleichtert die Ursachenforschung eines Fehlers zur Laufzeit

4 / 10

## Fallbeispiel

```
1. #include <stdio.h>
2.
3. void schreibeZahlen(int zahlen[], int anzahl)
4. {
5.     for (int i = 0; i < anzahl; i++)
6.     {
7.         zahlen[i] = i + 1;
8.     }
9. }
10.
11. void gibZahlAus(int zahlen[], int index)
12. {
13.     printf("%d\n", zahlen[index]);
14. }
15. void gibZahlenAus(int zahlen[], int anzahl)
16. {
17.     for (int i = anzahl; i >= 0; i--)
18.     {
19.         gibZahlAus(zahlen, i);
20.     }
21. }
22.
23. int main(int argc, char *argv[])
24. {
25.     int zahlen[10];
26.     schreibeZahlen(zahlen, 10);
27.     gibZahlenAus(zahlen, 10);
28.     exit(0);
29. }
```

5 / 10

## Was können wir jetzt tun?

Code hilfsweise bearbeiten

Testen, bis zu welcher Stelle der Code ausgeführt wird (mit `printf`)

Zwischenergebnisse ausgeben

hilfreich bei Fehlern unter besonderen Bedingungen

```
11. void gibZahlAus(int zahlen[], int index)
12. {
13.     printf("Zahl an Index %d:", index);
14.     printf("%d\n", zahlen[index]);
15. }
```

```
23. int main(int argc, char *argv[])
24. {
25.     int zahlen[10];
26.     schreibeZahlen(zahlen, 10);
27.     printf("Zahlen in Array geschrieben.\n");
28.     gibZahlenAus(zahlen, 10);
29.     gibZahlenAus(zahlen, 10);
30.     exit(0);
31. }
```

6 / 10

## Wie kann uns GDB an dieser Stelle helfen?

GDB „klingt“ sich in den internen Ablauf des Programms ein

Kompilieren: `gcc example.c -g -o example`

Ausführen: `gdb example`

Pausieren des Programms an beliebiger Stelle durch „Breakpoints“

`break [Datei].c:[Zeile]` oder `break my_func`

Einblick in RAM: Variablenwerte zur Laufzeit einsehen

`print my_var, watch my_var`

7 / 10

## Wie kann uns GDB an dieser Stelle helfen?

Zeilenweises Durchlaufen des Codes

`step, next, continue`

Stackframes untersuchen

`frame, up, down`

`backtrace`

8 / 10

## GDB in Entwicklungsumgebungen

GDB ist mit einer GUI sehr intuitiv zu bedienen

Integriert in: *Eclipse, Anjuta, Netbeans, Xcode, QT Creator*

Zusammenarbeit mit anderen Tools

*valgrind*

## Literatur

Wikipedia: „GNU Debugger“  
[https://en.wikipedia.org/wiki/GNU\\_Debugger](https://en.wikipedia.org/wiki/GNU_Debugger)

GDB: The GNU Project Debugger  
<http://www.gnu.org/software/gdb/documentation/>

Samuel Huang - GDB Tutorial  
<http://www.cs.umd.edu/~srhuang/teaching/cmsc212/gdb-tutorial-handout.pdf>

O'Reilly - Debuggen mit gdb  
<http://www.oreilly.de/german/freebooks/rlinux3ger/ch142.html>