D Fehlersuche mit Debugger

Ein Debugger dient dazu, Fehler in einem Programm zu finden, indem man es in seinem Ablauf beobachtet. Dazu kann man das Programm schrittweise ausführen lassen oder Haltepunkte (*breakpoints*) setzen und sich jeweils die Werte von Variablen anzeigen lassen. Auch der Ablauf selbst (Verzweigungen!) gibt manchmal Hinweise auf unerwartetes Verhalten.
Nach einem Absturz (z.B. *segmentation fault*) lässt sich anhand der Vorgeschichte (*backtrace*) feststellen, wo genau der Unfall geschah.
Voraussetzung ist, das man zunächst das Programm mit dem Schalter **-g** übersetzt – damit wird das Binärprogramm mit Informationen versehen, die Querverweise auf die Zeilennummern im Quellprogramms gestatten.

D.1 GDB

Der **GNU Debugger (GDB)** steht überall zur Verfügung. Gerade in seiner ursprünglichen Form (Kommandozeile, keine grafische Oberfläche) eignet er sich gut, um sich mit den Grundfunktionen des Debugging vertraut zu machen. Mit grafischen Debuggern findet man sich dann schneller zurecht.
In der Regel wird man beim **Aufruf** von GDB gleich das übersetzte Programm laden:

```
unix> gdb <executable>
...
(gdb)
```

In Tabelle 1 sind die wichtigsten Befehle zusammengestellt, ihre Benutzung in abgekürzter Form ist skizziert. Im Quellprogram orientiert man sich an der (angezeigten) Zeilennummer (`LNO`), die *breakpoints* haben eine eigene Nummerierung (`BNO`), ebenso die Ausdrücke für *display* (`DNO`).
Eine gut lesbare Einführung findet man z.B. in
`http://heather.cs.ucdavis.edu/~matloff/UnixAndC/CLanguage/Debug.html`

D.2 DDD

Anstatt die (alte) grafische Oberfläche `xxgdb` zu benutzen, sollte man den `GDB` unter `DDD` (*Data Display Debugger*) aufrufen:

```
unix> ddd <executable>
```

command	**usage**	
`help`	`h COMMAND`	explain a command
`file`	`file EXECUTABLE`	(re)load program

`list`	`l`
	`l LNO`
	`l SOURCE:LNO`
	`l FUNCTION`
		list source around current position,
		around specified line,
		in another source file,
		in the named function

`run`	`r` `r ARGUMENTS`	run program, with arguments and redirection
`step` `next`	`s` `n`	step program (one line), don't enter functions

`print`
`display`
 	
`undisplay`
	`p EXPR`
	`disp EXPR`
	`info disp`
	`undis DNO DNO ..`
	`undis`
		print contents of variable/expression
		print value at every interruption
		show display settings
		cancel specified displays
		cancel all displays

`break`	`b LNO`
	`b LNO CONDITION`
	`b FUNCTION`
		set breakpoint,
		with condition,
		at start of named function

`delete`
`disable`
`enable`
	`del BNO BNO ..`
	`dis BNO BNO ..`
	`en BNO BNO ..`
		delete / disable / enable
		specified break points
		(all if none specified)

`continue`	`c`	continue execution (after breakpoint)
`kill`	`k`	stop execution (e.g. to re-compile)
`backtrace`	`bt`	backtrace (after program crash)
`quit`	`q`	exit gdb

Tabelle 1: GDB – die wichtigsten Befehle.