INDEX

A
access errors, defined, 223
addresses, see also virtual memory
  addresses
  breakpoints, 68
arrays, see also dynamic arrays
  inspecting, 37
artificial arrays
  DDD, 107
  GDB, 106
assembly language, GDB and DDD,
  255–258
attributes, breakpoints, 67–69

B
backtrace, seg fault location, 4
binary search principle
  syntax errors, 189
  using, 4
binary trees, example, 95
brackets, matching, 208
breakpoints, 47–94, see also
  hardware-assisted breakpoints;
  temporary breakpoints
  about, 47
  command lists, 85–89
  conditional, 79–84
  DDD, 9
  deleting, disabling and moving,
    60–66
  GDB example, 56
  lists, 49
  persistence, 59
  resuming execution, 69–79
  setting, 51–56
  using, 14
  viewing attributes, 67–69
  watchpoints, 89–94
bugs, memory-access and seg faults,
  124
build scripts, libraries, 193
bus errors, 128
c
C
C language, see also GNU C library
  error reporting, 213–217
C99 library extensions, splint, 221
call command, 100
call stacks, moving up and down, 17
catchpoints
  defined, 55
  GDB, 48
CGDB, 13
client/server network programs,
  multiprogramming
  techniques, 145–151
command lists, breakpoints, 85–89
commands, see also specific commands
  DDD, 161
  Eclipse, 161
  GDB, 161
commands command, 99
compilers
  using, 212
  warnings, 210
compiling, 185–193
  missing libraries, 190
  multi-file programs, 57
  phantom line numbers in syntax
  error messages, 185–190
conditional breakpoints
  Eclipse, 42
  using, 79–84
conditional expressions, watchpoints,
  17
configure scripts, 193
confirmation principle
  about, 2
  GDB, 25, 31
  Java, 239
  resuming execution, 69
  syntax error messages, 187
continue command, 74
c
convenience variables
  names, 115
using, 114
core files
  crashes, 129–131
  seg fault location, 4
crashes, 117–144
  core files, 129–131
  example, 131–144
memory management, 118–129
critical sections, using, 156
curses programs, 194–201

D
data section, 119
DDD (Data Display Debugger)
  about, 2
  assembly language, 255–258
  breakpoint lists, 50
  command summary, 161
  conditional breakpoints, 83
  deleting and disabling breakpoints, 62
  versus Eclipse, 9
  example, 36–37
  inspecting variables, 102
  Java, 240–241
  moving breakpoints, 64
  Perl, 244
  Python, 249
  resuming execution, 78
  setting breakpoints, 55
  setting variables, 113
  undoing and redoing breakpoint actions, 66
  using GUI, 6
  viewing breakpoint attributes, 69
defaults, signal handlers, 126
deleting breakpoints, 60–64
Dijkstra algorithm, 172
disabling breakpoints, 62–64
display command, 98, 102
  displaying variables, 112
  disposition, breakpoints, 68
double frees
  defined, 224
  mtrace(), 231
dumping core, 129, 130
dynamic arrays, inspecting, 104
dynamically allocated memory,
  221–233
detecting problems, 224
  Electric Fence library, 225–228
  GNU C library tools, 228–233

E
Eclipse
  about, 2
  breakpoint lists, 51
  command summary, 161
  conditional breakpoints, 84
  versus DDD, 9
  deleting and disabling breakpoints, 63
  example, 38–43
  inspecting variables, 103
  Perl, 246–247
  Python, 250
  resuming execution, 79
  setting breakpoints, 56
  setting variables, 113
  using GUI, 7
  viewing breakpoint attributes, 69
Electric Fence library, dynamically allocated memory, 225–228
Emacs, features, 206
  enabling, breakpoints, 62
  errno, using, 213–217
errors, see also access errors; bus errors;
double frees; warnings
  compiler options for checking, 213
  reporting in C, 213–217
examples, see sample programs
exceptions, crashes, 128
execution, resuming, 69–79
expressions, see also conditional expressions
  library functions, 82
  watchpoints, 92, 93

F
faults, see segmentation faults
FIFO queue, example, 251
files, see core files; startup files
  compiling multi-file programs, 57
  finish command, 74
FPE (floating-point exception), 128
functions, see also specific functions
  string handling, 132
G
GDB (GNU Project Debugger), see also CGDB
about, 2
advantages, 11
assembly language, 255–258
breakpoint lists, 49
command summary, 161
conditional breakpoints, 80–83
deleting breakpoints, 61
disabling breakpoints, 62
debugging example, 22–36
expressions, 93
inspecting variables, 98
Java, 238–241
resuming execution, 70–78
setting breakpoints, 51
setting variables, 113
startup files, 43
TUI mode, 12
viewing breakpoint attributes, 67
.gdbinit files, 43
general protection fault, see segmentation fault
GNU C library
dynamically allocated memory, 228–233
GDB, 27
interrupts, 4
inspecting variables, 15
installing
DDD, 2
Eclipse, 2
integrated development environments (IDE), text editors as, 211
Intel stacks, 256
interfaces, text-based versus GUI, 5–14
interrupts, infinite loops, 4
Java, using GDB, DDD and Eclipse, 236–242
JDB (Java Debugger), DDD as GUI for, 241
L
layout, see program layout
lexical highlighting, defined, 206
libraries, see also C99 library extensions; curses programs;
Electric Fence library; GNU C library; static libraries
compiling missing, 190
NOW architectures, 170
SDSM, 170
types of, 191
library calls, errno, 213
library functions
compared to system calls, 217
GDB expressions, 82
line numbers, see phantom line numbers
lint, using, 219–221
Linux, dumping core, 130
lists, see also command lists
breakpoints, 49
loading, 185–193
missing libraries, 190
phantom line numbers in syntax
error messages, 185–190
local variables, monitoring, 112
loops, see infinite loops
ltrace, using, 217–219
M

makefiles

and compiler warnings, 210
and Vim, 209
MALLOCCHECK, 228
mcheck(), 230
memory, see also dynamically allocated
memory; virtual address space;
virtual memory addresses
examining directly, 112
memory leaks, mtrace, 231
memory management, crashes,
118–129
message passing, defined, 163
message-passing systems,
multiprogramming
techniques, 164–169
modular approach, see top-down
approach
monitoring, local variables, 112
moving, breakpoints in DDD, 64
mtrace(), 231
multiprogramming techniques,
145–183
client/server network programs,
145–151
example, 171–183
parallel applications, 163–171
threaded code, 151–163
muntrace(), 231

N

networks, multiprogramming

techniques for client/server
network programs, 145–151
non-int returning functions, 83
NOW architectures
libraries, 170

O

offsets, GDB, 53
online help
about, 19
OpenMP
example, 171–183
true shared memory, 170
operations, 14–18
inspecting variables, 15
moving up and down call stacks, 17
stepping through source code, 14
watchpoints, 17

P

page tables, 122
pages
about, 121
SDMS systems, 170
parallel applications,
multiprogramming
techniques, 163–171
parentheses, balancing, 208
Perl, DDD and Eclipse, 242–247
perror(), 216

phantom line numbers, syntax error
messages, 185–190
pkconfig program, 193
plain text, see text
principles, 2–5, see also binary search

principle; confirmation
principle; top-down approach
confirmation, 2
other, 4
printf(), using with trace code, 3
printing, variables, 112
process tables, defined, 152
processes, defined, 152
program layout, memory, 118
Pthreads, example, 151
Python, DDD and Eclipse, 247–251

R

redoing breakpoint actions in DDD, 66
reporting, see errors; warnings

S

sample programs
inspecting and setting variables, 95,
109
introductory debugging session,
19–43
seg faults, 131–144, 171–183
setting breakpoints with GDB, 56
threaded code, 153–161
saving symbol tables, 21

The Art of Debugging with GDB, DDD, and Eclipse
(C) 2008 by Norman Matloff and Peter Jay Salzman
SDSM (software distributed shared memory), libraries, 170
segmentation faults
  core files, 131
defined, 118
determining location, 4
  Eclipse, 42
  GDB, 32
memory-access bugs, 124
Unix signals, 125
setting
  breakpoints, 51–56
  variables, 113
  watchpoints, 90
shared memory, defined, 163
shared-memory systems
  example, 171–183
  multiprogramming techniques, 170
shells, core files, 130
Sieve of Eratosthenes, 153
signal handlers, Unix signals, 126
 signals
  seg faults, 125
Simplified Wrapper and Interface Generator (SWIG), using,
  251–254
snprintf(), 143
sockets, using, 148
software distributed shared memory (SDSM), libraries, 170
source code, stepping through, 14
splint, using, 220–221
stack frames, 17
stack sections, 119
stacks, see call stacks; Intel stacks
startup files, using, 45–45
static code checkers, lint and other tools, 219–221
static libraries, using, 191
stepping
  into versus over a function, 72
  through source code, 14
strace, using, 217–219
strerror(), 216
string handling, functions, 132
SWIG (Simplified Wrapper and Interface Generator), using,
  251–254
switches, splint, 221
symbol tables, saving, 21
syntax error messages, phantom line numbers, 185–190
syntax highlighting, text editors, 206
system calls, compared to library functions, 217
T
tables, see process tables; symbol tables
tasks, see processes
temporary breakpoints
  defined, 52
  Eclipse, 56
text editors, 206–212
  as IDEs, 211
  makefiles and compiler warnings, 210
  matching brackets, 208
  syntax highlighting, 206
  Vim and makefiles, 209
text section, 118
text, GDB, 6
versus GUI-based tools, 5–14
threads
  about, 151
  defined, 153
  multiprogramming techniques, 151–163
top-down approach
  about, 4
  GDB, 29
  stepping, 73
trace code, using, 3
trees, see binary trees
TUI mode, GDB, 12
U
ulimit command, 131
undoing breakpoint actions in DDD, 66
Unix
  signals and seg faults, 125
  virtual memory addresses, 118
Until command, 75
V
value history, using, 114

The Art of Debugging with GDB, DDD, and Eclipse
(C) 2008 by Norman Matloff and Peter Jay Salzman
variables, 95–115, see also convenience
variables; local variables
examing memory directly, 112
example, 95, 109
GDB’s own, 113
inspecting, 15
print and display, 112
setting, 113
watchpoints, 17
Vim
book about, 212
invoking make, 210
and makefiles, 209
syntax highlighting, 206
virtual address space, pages, 121
virtual memory addresses
breakpoints in GDB, 53
Unix, 118

W
warnings, see also errors
compiler options for, 212
compilers and makefiles, 210
splint, 220
watchpoints, see also hardware
watchpoints
binary search, 5
GDB, 48
using, 17, 89–94