GDB QUICK REFERENCE  GDB Version 4

Essential Commands
gdb program [core]  debug program  [using core dump core]
b [file]:function  set breakpoint at function  [in file]
run [arglist]  start your program  [with arglist]
bt  backtrace: display program stack
p expr  display the value of an expression
n  continue running your program
s  next line, stop after function calls
last  next line, stepping into function calls

Starting GDB
gdb  start GDB, with no debugging files
gdb program  begin debugging program
gdb program core  debug core dump core produced by program
-gdb help  describe command line options

Stopping GDB
quit  exit GDB; also q or ELF (e.g. C++-d)
INTERRUPT  [e.g. C++] terminate current command, or send to running process

Getting Help
help  list classes of commands
help class  classwide descriptions for commands in class
help command  describe command

Executing your Program
run arglist  start your program with arglist
run  start your program with current arguments
run <inf>stop  start your program with input, output redirected
kill  kill running program
tty  use tty as stdin and stdout for next run
set args arglist  supply arguments to next run
set args  specify empty argument list
show args  display argument list
show env  show all environment variables
show env arg  show value of environment variable arg
set env var  set environment variable var
unset env var  remove var from environment

Shell Commands
cd dir  change working directory to dir
pwd  Print working directory
make ...  call `make`
shell cmd  execute arbitrary shell command string

Breakpoints and Watchpoints
break [file]:line  set breakpoint at line number [in file]
b [file]:line  eg. break main.c:37
break [file]:func  set breakpoint at func [in file]
break offset  set break at offset lines from current line
break offset  set breakpoint at offset
break addr  set breakpoint at address addr
break . . . if expr  break conditionally on non-zero expr
break cond [expr]  new conditional expression on breakpoint
 cond if expr  make unconditional if expr 
 break . . . if expr  temporary break, disable when reached
break reg expr  set a watch expression on expression
watch expr  set a watch expression
catch r  break when C++ handler for expression r
info break  show defined breakpoints
info watch  show defined watchpoints
info clear  delete breakpoints at next instruction
info clear [file]:func  delete breakpoints at entry to func()
info clear [file]:line  delete breakpoints on source line
info delete [n]  delete breakpoints [or breakpoint n]
disable [n]  disable breakpoints [or breakpoint n]
enable [n]  enable breakpoints [or breakpoint n]
able once [n]  disable again when reached
able del [n]  delete breakpoints [or breakpoint n]
info ignore n count  ignore breakpoint n, count times
info commands [silence] commandlist  execute GDB commandlist every time
breakpoint n is reached. [silence]
commandlist  end of commandlist

Program Stack
backtrace [n]  print trace of all frames in stack; or of a frame—innermost first, outermost last
bt [n]  set bt [n]  display backtrace
frame [n]  select frame number n or frame at address n; if no n, display current frame
up n  select frame n frames up
down n  select frame n frames down
info frame [addr]  describe selected frame, or frame at addr
info args  arguments of selected frame
info locals [addr]  local variables of selected frame
info reg [reg]  register values [or registers reg] in selected frame
info all-reg [reg]  register values [or registers] in selected frame
info catch  exception handlers active in selected frame

Execution Control
continue [count]  continue running; if count specified, ignore c [count]
step [count]  execute until another line reached; repeat s [count] times if specified
stop [count]  step by machine instructions rather than source lines
next [count]  execute next line, including any function calls
nexti [count]  next machine instruction rather than source line
until location  run until next instruction (or location)
finish  run until selected stack frame returns
return [expr]  return from selected stack frame without executing [return value]
signal name  resume execution with signal n (none if 0)
signal name [count]  resume execution at specified line number
jump line  or address  evaluate expr without displaying it; use for altering program variables

Display
print [file] [expr]  show value of expr [or last value $]
p [file] [expr]  according to format $:
x  hexadecimal
u  unsigned decimal
t  octal
b  binary
a  address, absolute and relative
w  character
f  floating point
call [file] [expr]  like print but does not display void
x [Nw] [expr]  examine memory at address expr; optional format spec follows slash
N  count of how many units to display
n  unit size; one of
1  individual bytes
h  halfwords (two bytes)
w  words (four bytes)
g  count of how many units to display
j  giant words (eight bytes)

printing format. Any print format, or
/t  multi-terminated string
i  machine instructions
disassem [addr]  display memory as machine instructions

Automatic Display
display [file] [expr]  show value of expr each time program
steps according to format $
display  display all enabled expressions on list
undisplay  remove number(s) n from list of
disable disp n  automatically displayed expressions
enable disp n  display for expression(s) number n
info display  number list of display expressions

[[[]]]  surround optional arguments  ... show one or more arguments
