GDB GNU debugger

CONCEPTS

Compile with -g (and -O0 for sane stepping), otherwise GDB shows no source lines or local variables.

Break break, watch
↓ `run`
Stopped frame, locals
↓ `step / continue`
Inspect print, x, info
↓ `repeat`
Resume next, finish, continue

START AND ATTACH

`gdb <prog>`
Load a program

`gdb --args <prog> <args>`
Load a program with arguments

`gdb -p <pid>`
Attach to a running process

`gdb <prog> <core>`
Inspect a core dump

`run [args]`
Start the program

`start`
Run and stop at the start of main

BREAKPOINTS

`break <func>`
Break at a function

`break <file>:<line>`
Break at a line of a file

`break <loc> if <cond>`
Break only when the condition holds

`tbreak <loc>`
One-shot breakpoint, deleted after first stop

`info breakpoints`
List breakpoints and watchpoints

`delete [<n>]`
Delete one breakpoint, or all

BREAKPOINTS (cont.)

`disable <n> / enable <n>`
Turn a breakpoint off / on

WATCHPOINTS

`watch <var>`
Stop when a value changes

RUN CONTROL

`continue [<n>]`
Resume; optionally skip n-1 further hits here

`next [<n>]`
Step over calls to the next line

`step [<n>]`
Step into functions with debug info

`finish`
Run until the current function returns

`Ctrl+C`
Interrupt the running program

STACK

`backtrace`
Show the call stack

`up [<n>] / down [<n>]`
Move toward the caller / callee

`info locals`
Local variables of the frame

INSPECT DATA

`print <expr>`
Evaluate and show an expression

`print/x <expr>`
Print in hex (also d, o, t, c, a)

`x/<n><f><u> <addr>`
Examine memory

TEXT UI

`layout src`
Source window above the command line