CodeQuestHub.io - GDB Cheat Sheet

Starting / Stopping / Attaching

`gdb <program>` - Start GDB with a program
`gdb -p <pid>` - Attach to a running process
`gdb <program> <core>` - Load a core dump
`attach <pid>` - Attach to a PID
`set args <args>` - Set program arguments
`run` - Run the program
`start` - Run until main()
`kill` - Send the kill signal
`detach` - Detach from the process

Printing / Inspecting State

`print <expr>` - Evaluate and print expression
`display <expr>` - Display expression on every loop
`info locals` - Show local variables
`info args` - Show function arguments
`info registers` - Show CPU registers
`x/<format> <address>` - Examine memory
`set var <var>=<value>` - Set a variable value
`disassemble` - Disassemble a function
`info variables` - Show global and static variables

Stack Traces and Info

`backtrace` - Show call stack
`where` - Alias for backtrace
`frame <n>` - Select stack frame
`up` / `down` - Move up/down one frame
`info threads` - Show threads
`info breakpoints` - Show breakpoints
`info files` - Show loaded files
`info sharedlibrary` - List loaded shared libraries
`whatis <var>` - Show type of variable

Breakpoints / Navigation

`break <function>` - Set breakpoint at `function`
`break <file>:<line>` - Set breakpoint at `file:line`
`tbreak <function>` - Temporary breakpoint
`delete <n>` - Delete breakpoint number n
`disable <n>` - Disable breakpoint number n
`enable <n>` - Enable breakpoint number n
`continue` - Continue running after breakpoint
`step` - Step into function call
`next` - Step over function call
`finish` - Run until current function returns
`watch <expr>` - Break when expression written
`rwatch <expr>` - Break when expression read
`awatch <expr>` - Break when expression accessed
`break <loc> if <cond>` - Conditional breakpoint
`condition <n> <expr>` - Set condition on breakpoint
`commands <n>` - Set commands to run at breakpoint `n`
`ignore <n> <count>` - Skip breakpoint `n` `count` times

Reverse Debugging

`record` - Start recording execution
`record stop` - Stop recording
`reverse-stepi` - Step backward one instruction
`reverse-continue` - Continue backward to breakpoint

Signals

`info signals` - List all signals and handling
`handle <signal> <actions>` - Set signal handling
`signal <signal>` - Deliver signal manually
`catch <signal>` - Break when a signal is raised

Memory Display (x Command) Format and Examples

`b`	Byte (1 byte)	`x/4xb $esp`	4 bytes at stack pointer, hex
`h`	Half word (2 bytes)	`x/8xh $esp`	8 half words at stack pointer, hex
`w`	Word (4 bytes)	`x/2xw 0x61050`	2 words at address 0x61050, hex
`g`	Giant word (8 bytes)	`x/1xg $rbp`	1 giant word at frame pointer, hex
`c`	Char	`x/10cb $esp`	10 bytes at stack pointer, as chars
`d`	Signed decimal	`x/6dw 0x400600`	6 words as signed decimals
`u`	Unsigned decimal	`x/4uw $esp`	4 words as unsigned decimals
`x`	Hexadecimal	`x/4xw $esp`	4 words as hex
`o`	Octal	`x/4ow $esp`	4 words as octal
`t`	Binary	`x/5tb $esp`	5 bytes as binary
`s`	C String	`x/s 0x601000`	View memory as C string
`a`	Address pointer	`x/a $rbp`	View address at base pointer