Debuggers

- Step through code one line at a time
- Inspect variables, including structs and classes
- View disassembly
- Check the call stack

Debuggers

- Step through code one line at a time
- Inspect variables, including structs and classes
- View disassembly
- Check the call stack
- `gdb` Command-line debugger
- `kdbg` GUI frontend for `gdb`

Using `gdb`

- `gdb your-program` launches the debugger
- Note: You will want to compile with `g++ -g`
- `run arg1 arg2 ...` runs the command with command line arguments
- `backtrace` or `bt` shows the call stack when the program terminates

Looking at variables with `gdb`

- `p variable` prints the contents of ‘variable’.
- `p` also works with expressions of just about any sort.

Looking at variables with `gdb`

- `p variable` prints the contents of ‘variable’.
- `p` also works with expressions of just about any sort.
- `x address` examines one word memory at a given address.
- `x/2 address` examines two words of memory.
- More on examining memory

Setting breakpoints with `gdb`

- `break filename.cpp:10` will stop execution whenever line 10 in ‘filename.cpp’ is reached.
- `continue` resumes running as normal.
- `step` runs one more line of code.
- `next` runs until execution is on the next line.
- `finish` runs until the current function returns.
- `delete` removes all breakpoints.
- More on breakpoints

Using `kdbg`

- Remember to enable X forwarding
- `kdbg your-program` launches the debugger
- ‘Stack’ window: the function call stack
- ‘Locals’ window: variables currently in scope
- ‘Watches’ window: list of expressions (like ‘print’)
- f2 or right-click to edit the value of a local or watch!

Breakpoints in `kdbg`

- Set a breakpoint by clicking in the margin left of the code
- f5 Run (combination of 'run' and 'continue')
- f8 Step Into (like 'step')
- f10 Step Over (like 'next')
- f6 Step Out (like 'finish')

Breakpoints in `kdbg`

- Set a breakpoint by clicking in the margin left of the code
- f5 Run (combination of ‘run’ and ‘continue’)
- f8 Step Into (like ‘step’)
- f10 Step Over (like ‘next’)
- f6 Step Out (like ‘finish’)
- ‘Breakpoints’ window: list of breakpoints
- Conditional Breakpoints: Set a condition on the breakpoints tab

Looking at program internals with `kdbg`

- Click the ‘+’ to see disassembly!
- ‘Memory’ tab: View memory at a specific address (right-click for display options)
- ‘Registers’ tab: View contents of CPU registers
- ‘Step into by instruction’ and ‘Step over by instruction’