Debugging with GDB

- GDB -- GNU DeBugger
- On lab machines and ldc.cs.wwu.edu machines

Compiling
- gcc -g hello.c
- gcc -g -o hello hello.c

Running
- Normal run: no difference
- Under gdb:
  - gdb a.out
  - gdb hello
- ddd or xxgdb:
  - ddd a.out
  - xxgdb hello
Basic GDB Commands

- Help: `help`
- Breakpoints: `break main`
- Running: `run arg_list`
- Step to next line: `next`
- Step into functions: `step`
- Continue running: `cont`
- List source: `list`
- Quiting: `quit`

Running GDB with a core dump ... (p1.c)
- NetBSD (possibly other *BSDs)
  - `gdb a.out a.out.core`
  - `gdb name name.core`
- Linux, Solaris & others
  - `gdb a.out core`
  - `gdb name core`
- Having Linux generate a core dump (bash shell)
  - `show all limits`: `ulimit -a`
  - `set core limit`: `ulimit -c unlimited`
GDB commands useful with a core dump

- location & func calls -- where
- printing variables
  - info locals
  - info args
  - info frame
- up
- down
- print var_name
- print /o var_name
- print /x var_name
- printf "formatstring", data, data, ....
- printf "a=%d\n", a
Infinite loops

- ps(1) -- get pid
- attach pid
- set variable name = value
- detach
- kill
Functions (p3.c)

- step vs next
- print f(n)
- call f(n)
- where
- finish
Breakpoints -- stopping in time! (funcs.c)

- Setting
  - `break function_name`
  - `break`

- Listing breakpoints: `info break`
- Disable breakpoint: `disable 24`
- Enable breakpoint: `enable 24`
- Deleting: `delete break num`
- Conditional: `break <pos> if <expr>`
- Printing at breaks: `display expr`
- Listing displays: `info display`
- Not printing: `undisplay n`
GDB Variables
- set $i = 0
- print a[$i++] (over and over again [cr])
- printf "a[%d] = %d\n", $i, a[$i++]

Printing dynamic arrays (dyn.c)
- print *a@len

User Defined commands
- define name
- ...
- end
- $arg0, $arg1, ... $arg9
- if/else/end
- while/end
- document command
GDB initialization

- ~/.gdbinit

Source code

- list
- search
- reverse-search
- dir
- show directories
- info sources
- info functions
- info variables
Global variables

global.c:

run it

set write on
file a.out
set var debug = 0
set var x = 20
quit

run it again

Note: int debug = 0; -> in BSS and doesn't work.
Commands at a breakpoint (bug.c)

break ...
commands
....
end

☐ silent
☐ continue

☐ if statements!