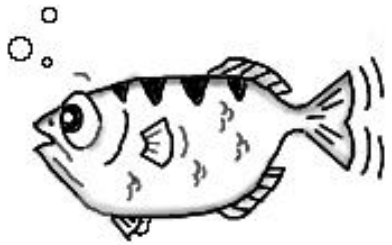
GDB

The GNU Project Debugger

Concurrent Programming

What is GDB?

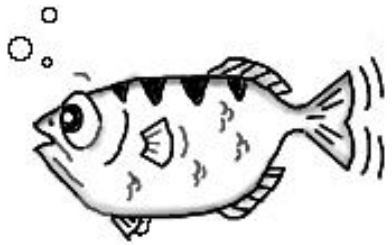
“GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed.”



What is GDB?

“GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed.”

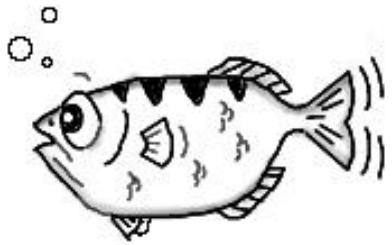
- Start your program, specifying anything that might affect its behavior.



What is GDB?

“GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed.”

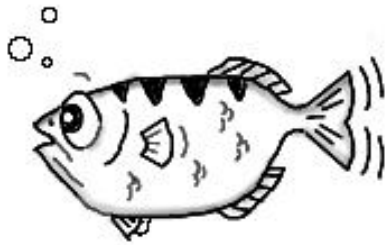
- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.



What is GDB?

"GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed."

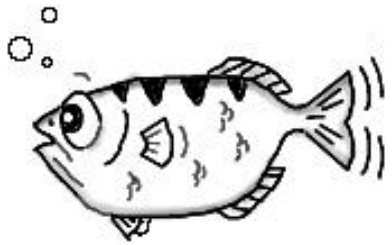
- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.



What is GDB?

"GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed."

- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.
- Change things in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.



How to use?

- Compile with -g option

```
$ g++ -g prime_mt_bug.cpp -o prime_mt_bug -lpthread
```

- Run gdb

```
$ gdb ./prime_mt_bug
```

- Lots of commands, lots of resources
  - https://www.gnu.org/software/gdb/

Useful commands

- Run (r)
  - runs the program (with arglist, if specified)
- Quit (q)
  - quit gdb

Useful commands

- Break (b)
  - Set a breakpoint

```
(gdb) b main
(gdb) b 25
(gdb) b prime.cpp:25
(gdb) b prime.cpp:25 if n > 100000 && n < 200000
(gdb) b prime.cpp:25 thread 7
```

- Info b
  - Display the breakpoint list
- Delete [#]
  - Delete a breakpoint #

Useful commands

- List (l)
  - Print 10 following lines of code

```
(gdb) l
(gdb) l 25
```

- Next (n)
  - Execute next program line. Step *over* any function calls in the line
- Print (p) [*variable_name*]
  - Print value of the

Useful commands

- Info thread
  - Display the running threads

- Thread [#]
  - Switch to thread #

Useful commands

- Step (s)
  - Execute next program line. Step *into* any function calls in the line
- Finish (fin)
  - Execute until selected stack frame returns
- Backtrace (bt)
  - Display the program stack
- Up
  - Move a frame up the stack
- Down
  - Move a frame down the stack

Practice

- Download *prime_mt_bug.cpp* from the Piazza resource page

Thank You