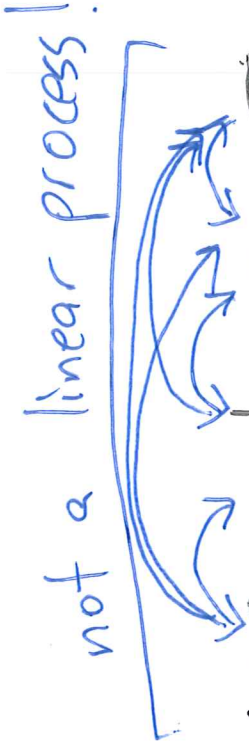
Objectives - Tue 1/29/2019

- Quiz 1: Address syntax
- Assertions
  - `#include <assert.h>`
    `assert(…)`
  - Verify and document assumptions
  - Find bugs earlier, right at the source
  - For finding bugs in your code only
  - Not for checking network status, existence of file, validity arguments passed from external code, etc.
- GDB - intermediate debugging
  - Watchpoints – watch
  - Reverse debugging – record full, reverse-step, …
  - How to learn about other commands – help

Time to develop a program



Understand requirements: ___ hours

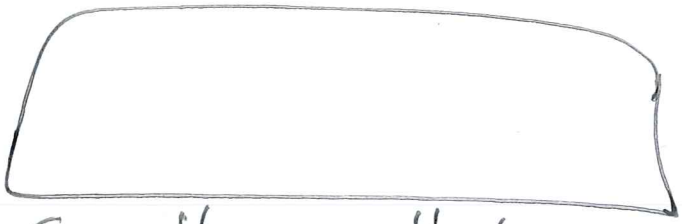
Write code: ___ hours

Test: ___ hours

Debug: ___ hours ← most volatile, most control

You can control the time you spend debugging by testing proactively, using test-driven development (TDD) and defensive coding strategies (→ code quality)

#include <assert.h>

assert()

Something that must be true unless you have a bug in your code

(assert is a macro)

- Check assumptions about your code.
- Document assumptions while checking.
- Helps you find malfunction at its source. ↑ In lieu of a comment
- Narrow the search space (where bug might be)
- Code tests itself (at least a little)

⚠ Don't use assert(...) to

- check valid inputs
- check system state
- check if file exists
- assert(remove_file(-))
- assert(n++);

} Inside action won't happen in production

To eliminate assertions: ← Never needed in ECE 264.

gcc -DNDEBUG ~~~