David Agans’ *Debugging*

- Short book on general principles of debugging
- Structured around a set of simple rules that really are a good idea
David Agans’ *Debugging* Rules

- Rule #1: “Understand the System”
  - “READ THE MANUAL”

Debugging something you don’t understand is pointlessly hard

Just as with testing, subject knowledge matters – here you need knowledge of the source code as well
David Agans’ Debugging Rules

- Rule #2: “Make It Fail”
- You can’t debug what you can’t produce
- Find a way to reliably make a system fail
- Record everything, and look for correlation
  - Don’t assume something “can’t” be a cause
David Agans’ Debugging Rules

- Rule #3: “Quit Thinking and Look”

- Don’t hypothesize before examining the failure in detail – examine the evidence, then think

- Engineers like to think, don’t like to look nearly as much (instrumentation and running a debugger both look like work)

- “If it is doing X, must be Y” – maybe
  - Check
David Agans’ *Debugging* Rules

- **Rule #4:**
  
  “Divide and Conquer”

- *This rule is the heart of debugging*
  
  - Heart of delta-debugging
  - Narrow down the source of the problem
  - “Does it still fail if this factor is removed?”
  - Use a debugger to check system state at checkpoints; if everything is ok, you’re before the problem
David Agans’ *Debugging* Rules

- **Rule #5:** “Change One Thing at a Time”

- A common very bad debugging strategy:
  - “It could be one of X, Y, Z. I’ll change all three, and run it again.”

- Isolate factors, because that’s how you get experiments that tell you something

- If code worked before last checkin, maybe you should look at *just those changes*
David Agans’ *Debugging* Rules

- Rule #6: “Keep an Audit Trail”
  
- Don’t rely on your perfect memory to remember everything you tried
  
- Don’t assume only you will ever work on this problem
  
- Rule of thumb: 20 minutes
David Agans’ *Debugging* Rules

- Rule #7: “Check the Plug”

- Question assumptions
- Don’t always trust the debugger
- Don’t trust your tests
David Agans’ *Debugging* Rules

- Rule #8: “Get a Fresh View”
  - It’s ok to ask for help
  - Experts can be useful
  - Explain what happens, not what you think is going on
David Agans’ *Debugging* Rules

- **Rule #9:** “If You Didn’t Fix It, It Ain’t Fixed”
  - Once you “find the cause of a bug” confirm that changing the cause actually removes the effect
  - A bug isn’t done until the fix is in place and confirmed to actually fix the problem
    - You might have just understood a symptom, not the underlying problem