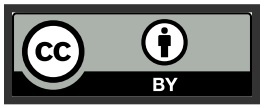
NOT ON THE SHELVES

GREG WILSON

gvwilson@third-bit.com

NOVEMBER 2016



Version 1.3

Once Upon a Time...

- Support programmer in university computing center
- Saw smart people stumble because ***no one taught them basic skills***

- First Software Carpentry class ran at LANL in 1998

Time Passes...

- Book review editor for *Doctor Dobb's Journal*
- Hundreds of textbooks on compilers, but ***no textbooks on debuggers or debugging***
- Or build tools, or package managers, or...

More Time Passes...

- Asked to teach a course on software architecture
- Looked at two dozen books and other people's courses...
- ...but ***no textbooks describe actual architectures***

How Learning Works

- Started reading the education literature in 2011
- We know a lot about learning and teaching
- But ***most faculty have never been taught how to teach***

We Can't Get There From Here

- "Computer Science for All" is a great rallying cry...
- ...but ***most programmers have never been taught how to teach*** either

Computer ~~Science~~ Strong Opinion

- We know a lot about software and how it's built
- But ***students aren't taught empirical methods***
  - Biologists spend 6 hours/week in the lab
  - CS students do one experiment in four years

There's no shortage of material

Machines Learn What We Teach Them

- "Machine learning is money laundering for bias"
- Because ***bias isn't part of the discussion of algorithms***
- Just as ***harassment isn't part of the discussion of distributed systems***
- Same cause: ***programmers aren't taught how to empathize***

What's the Pattern?

One-way flow of information

"We" will talk, "they" will listen

(for many values of "they")

Proposal #1: Teach Critical Analysis

- Software engineering courses don't take advantage of the billions of lines of software that are now openly available
- Or capitalize on the current craze for data science

So let's do that.

Proposal #1: Teach Critical Analysis

Given version control repositories for six software projects, determine whether long functions and methods are more likely to be buggy than short ones.

- Requires tool use, model building, and statistics
- Encourages students to *do*, so they *understand and value*, so they *engage*
- Fits into existing curriculum
- And it's *culturally defensible*

Proposal #2: Teach Teaching

- Create programs in CS education modelled on those in math and physics education
- Create the teachers our schools need…
- …and the students we wish we had

But who will teach the teachers?

Proposal #2 (revised): Teach Teaching

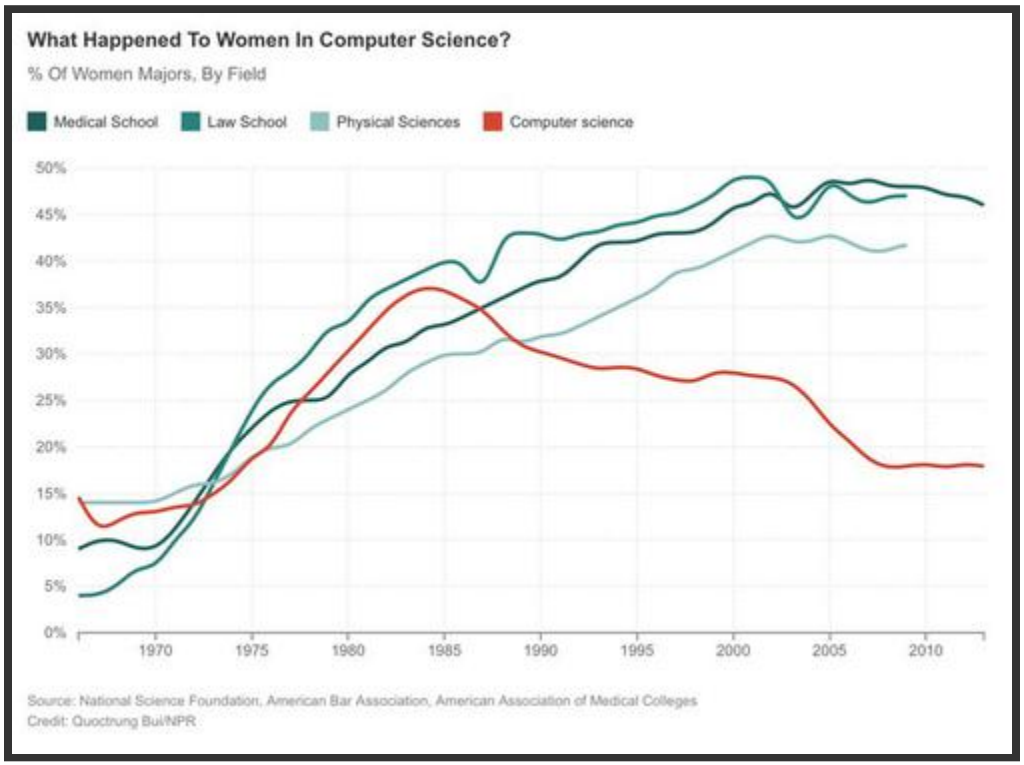
Start with a single course

- Ambrose et al: *How Learning Works*
- Lemov: *Teach Like a Champion*
- Guzdial: *Learner-Centered Design of Computing Education*
- ...and dozens of papers from the last thirty years

They'll be doing science in this course too.

Bait and Switch

- I no longer believe that we will fix this



- But we can raise a generation that will

Teach Empathy

- Margolis and Fisher: *Unlocking the Clubhouse*
- Margolis: *Stuck in the Shallow End*
- Schneier: *Data and Goliath*
- Jeong: *The Internet of Garbage*

And everything else that we pretend isn't our responsibility

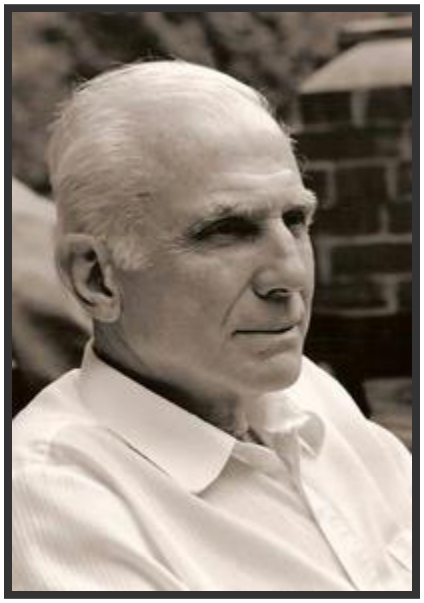
Teach Empathy

1. *People of East Asian or South Asian ancestry make up 8% of the general population, but 60-75% of undergraduates in Computer Science at major universities. Write two 1000-word position papers to argue pro and con the proposition that this proves people of European descent are naturally less capable of abstract reasoning than their Asian counterparts.*
2. *Compare and contrast your arguments with those made about female under-representation in computing.*

A Sudden Sense of Urgency

Volume 6: What Everyone In Tech Absolutely, Positively Must Know About Racism, Sexism, Economics, Poverty, Harassment, Privacy, and a Whole Lot More

Want to contribute? gvwilson@third-bit.com



Thank you.

gvwilson@third-bit.com