Educating Programmers: A Customer Perspective

Alexander Stepanov
A9.com

**Workshop on Quality Software:
A Festschrift for Bjarne Stroustrup
Texas A&M University, April 27-28, 2012**

Abstract

Many programmers lack the skills required for producing quality software. Programming must be taught as a serious discipline, with an extensive core curriculum covering such topics as system decomposition and component design. Developing a sense of programming aesthetics, including the study of real world examples of beautiful programs, is at the heart of this discipline. Grounding in elementary mathematics (algebra and Euclidean geometry) provides the necessary intellectual and aesthetic foundation for the curriculum.

Background

For the last 17 years I have been trying to improve software quality at several companies:

- Silicon Graphics
- Adobe
- A9.com

All leading-edge companies, employing graduates of leading universities.

I worked with different teams, studied their code, and taught advanced classes.

Observation

- Everybody works as a programmer.
- Nobody really knows how to program.
- The idea that there is something more to learn does not even cross their minds.
  - If they want to learn, it is a new language or a new tool: Java, Hadoop, Squid, etc.

Good code

- Useful interface, efficient implementation, pleasure to read
  - Yes, it should be enjoyable to read code
- If X is original development time
  - Takes very little time (< 1% X) to learn to use
  - Takes little time (< 10% X) to extend and modify
- Good is beautiful, beautiful is good

Beautiful Code is for Toy Examples Only?

"Real programs are always ugly..."

– a well-known computer scientist and textbook author

Fundamental Programming Skills

- Architect systems from components
- Design correct, consistent, extensible APIs
- Recognize known abstractions
- Know how and when to make code efficient

Computer Science for Programmers

- Computer Architecture
  - characteristics of CPUs, not their design
- Compiler
  - understanding its limitations
  - utilizing its features
- OS
  - using threads, not writing schedulers
- Databases
  - how and when to use SQL, not design relational DB

What about Programming?

- Today only introductory programming is taught.
- No real programs are studied.
- Testing and measurement techniques are not taught.

What should be done?

A Programming Curriculum

- Introductory programming
  - Control structures and basic data structures
- Intermediate programming
  - Using advanced components and tuning them
- Advanced programming
  - Designing new components
- Master class
  - Building a system out of components

Writing

- A class where programmers are taught to describe software.

- A programmer must be able to communicate their design to others.

Programming in Academia

- Programming should be taught by people who know and love programming and have done it for a living
- Nemo dat quod non habet
  - Music departments

An Internet Journal of Programming

- To publish refereed
  - components
  - measurements
  - validation
  - testimonials
  - use cases

Grounding in Mathematics

- Mathematics has served every scientific and engineering discipline for many centuries.
- Mathematics is the science of the abstract.
- Rich mathematical heritage helps to develop programming aesthetics.

Which Mathematics?

- Elementary Algebra
  - Scope of George Chrystal
    - e.g., continued fractions, Bernoulli numbers, etc.
    - supplemented with a little abstract algebra
  - Develops symbol manipulation and abstraction ability
- Euclidean Geometry
  - Euclid is still the best
    - coherent story from book I to book XIII
  - Develops architectural and reasoning ability

Conclusion: Programming Aesthetics

- Sense of beauty is important for building large systems and controlling complexity.

- Study of mathematics develops this sense.