David W. Proctor

Student, Yale University, Class of 2019 ~ www.daveyproctor.com

Education

YALE UNIVERSITY
Timothy Dwight
Class of 2019
New Haven, CT
B.S. Computer Science (Prospective)

HOPKINS SCHOOL
Class of 2015
New Haven, CT

Coursework

Algorithms
Data Structures
Discrete Math
Quantum Computing
Computational tools for Astrophysics

Skills

Programming
C
Python

Data
Monte Carlo error estimation
Chi^2 curve and template fitting
Noise reduction

Language
Spanish
Portuguese

Experience

CS50 | Developer
2017 summer. cs50.net
Harvard and Yale's "Intensive Introduction to Computer Science."
• Led three technical presentations at each of two CS50 educator conferences.
• Taught two-week web development course (Python, JS, SQL) at Harvard Summer School.
• Made CS50 library install cross-platform (Mac, Linux).
• Analyzed teacher and student survey data.
• Created reference sheets for core topics. ap.cs50.net/resources.
• Designed cross-curricula (physics, astronomy) lesson plans and psets.

CS50 | Teaching Fellow
2016 fall. ap.cs50.net
I hosted section, office hours and graded 18 Harvard Extension School students.

CS50 AP | Curriculum Intern
2016 summer.
In 2016-17, CS50 AP was taught in 40+ high schools to 1000+ students.
• Mapped CS50 materials to a high school AP curriculum.
• Developed two Python web apps (psets 7 and 8).
• Generated questions for CS50 Quizbank (quizbank.cs50.net), a student's study tool and teacher's quiz generator.

Honduras Childrens Project | President
2013-present. honduraschildrensproject.org.
Honduras Children's Project is "dedicated to improving the education for the children of Copprome Orphanage

daveyproctor.com github.com/daveyproctor linkedin.com/in/daveyproctor david.proctor@yale.edu (860) 754-7150

CS50 Scalability

Key Terms
* Horizontal scaling
* Vertical scaling

Overview
Scalability refers to a program's ability to grow, be it in terms of resource usage, input size, or general workload. Scalability can take many specific forms, depending on the context; for instance, making a scalable web service would entail different considerations than making a scalable C program.

Web Scaling
In the context of web services, there are two main subsets of scaling: **horizontal** and **vertical** scaling. Horizontal scaling refers to the act of scaling via expanding the *number of machines* a system uses. For instance, Facebook could increase the number of users they can support by simply buying more servers. Vertical scaling, meanwhile, involves doubling down and investing in *better components for the current machines*. For example, you could upgrade the hard drive or CPU for the servers you already have. Both of these have their potential upsides and downsides. For example, horizontal scaling presents an issue of routing: how do we connect multiple servers to a single DNS? Vertical scaling, meanwhile, makes the website solely reliant on one server: if the one machine fails, the website has no backup. In practice, both techniques are used in tandem to overcome each technique's respective shortcomings.

Lower-level Scalability
Scalability isn't solely of importance in modern web applications, however. Think of some practices we insisted that you follow when writing C programs---remember `sizeof`? Using this function in fact adds scalability to your programs. Without it, the program is architecture-dependent. Some datatypes in C, such as `int`, are sized according not to the C standard but to suit the architecture of the computer upon which you're working. For example, an `int` on a 32-bit machine would be 4 bytes (4 B = 32 b) but on a 64-bit machine (like most modern computers are) could be 8 bytes. As a result, using `sizeof` is vital, especially if you happen to be working on a 32-bit machine and hoping for people with 64-bit machines to run your program!
Note that not all scalability is as easy as using a function, unfortunately! Try again to think back over some of the problems that you've done for CS50 AP. Were there any assumptions we allowed you to make about, say, user input? Putting a maximum size on the amount of user input that will be taken often times makes implementing a program dramatically easier, as it changes it from a problem which requires dynamic memory allocation to static. Not only do you have to use pointers (ugh!) and the like in order to accept arbitrarily-sized inputs, but you have to handle any new errors that can arise in that context as well. I guess that's what they call "growing pains!"

daveyproctor.com
github.com/daveyproctor
linkedin.com/in/daveyproctor
david.proctor@yale.edu
(860) 754-7150

Coursework

Computer Science
Algorithms
Data Structures
Discrete Math

Physics
Quantum Computing
Computational tools for Astro-

Key Terms

- Horizontal scaling
- Vertical scaling