Davor Cubranic
Scientific Programmer
Department of Statistics
University of British Columbia

September 2, 2011

Josh Greenberg
Alfred P. Sloan Foundation
630 Fifth Avenue
Suite 2550
New York, NY, 10111

Dear Mr. Greenberg:

In my role as scientific programmer in the Department of Statistics, I am regularly exposed to code written by graduate students with little training in programming and no exposure to good software development practices. My PhD research was in software engineering and I worked for six years in the industry, so one of my goals in taking on my current job has been to learn more about how non-professionals write code and approach computational problem-solving. Despite my being familiar with challenges scientists face when writing software for their research (largely through academic publications by Dr. Wilson and others), I can honestly say that it has been quite an eye-opening experience.

When our students are under pressure to get on with their research and are surrounded by other students who are similarly unsophisticated as software developers, they end up working inefficiently and create inefficient code. Tools that have been around for 30 years or more—like debuggers, test frameworks, or version control—are not even known about, much less used. Thus, programs students create as part of their research take longer to write and run slower and less reliably than they should. This bad legacy continues after their creator has graduated and left university, leaving behind code that is difficult to use and maintain, and making it challenging for others to build on their work.

Everybody knows that this is a problem, and all professors tell me that they would like their students to be better programmers. Currently, there are very limited avenues to make this happen within our institution. I try to do my part and regularly put on workshops on essential programming tools or technologies. Students attend and are interested in how to improve their programming, but attending a two-hour workshop is not how these skills are learned. An official, semester-long course is another option, but including it into a graduate degree program is a difficult slog through the university bureaucracy and something that does not happen quickly.

For these reasons, I am excited by the content and format of the proposed course put forward by the Mozilla Foundation and Software Carpentry. I think its combination of tutorials, hands-on coding, and building of online learning communities is a very promising way of spreading the knowledge, skills, and *culture* of good scientific software development. Just as importantly, the course acknowledges institutional realities and works within the existing system to provide an attractive learning option to all the parties involved. I hope to work with the team to make this option available to students at my university.

Sincerely,

Davor Cubranic, PhD