Software Skills Training, Inc.

Did a programmer just kill 20 Indians?

When software engineers look for quick and easy answers, the results are sometimes less than ideal.

The Cost of Easy Answers: Disasters Caused by Amateurish Programming

Recently, a bit of sloppy programming appears to have caused at least 20 suicides in India. It seems that in the Indian educational system, test results have a profound effect on the future of students, since India's higher education system does not have the capacity to educate all the students who are qualified. A suboptimal score eliminates students from consideration for admittance to many institutions, and students who fail to achieve the necessary scores, right or wrong, often feel they have no future. So when, of the 1,000,000 tested, 350,000 -- a number that was far higher than expected, that was unlike any previous results, and that was, it turns out, wrong -- were told, because of a programming error, that they had failed, at least 20 students committed suicide.

And that's not all.

There are many similar cases. In 1990, over 50 percent of AT&T's network crashed, and 75 million calls failed to connect, because of a standard software update that had an undetected bug. We have no idea how many emergency calls went unanswered and how many died as a result.

In 1978, the Hartford Coliseum collapsed due to a glitch in the CAD software used to design the roof.

There are endless cases where code intended for debugging on ecommerce websites got mixed up with production code, and had the result of confusing customers and costing sales from hundreds to thousands of dollars.

WHY??

Clearly, one of the major culprits is the phenomenon of *Quick-Answer Programming*, the practice of having untrained or minimally trained "programmers" develop code. All too often, they figure out how to solve a particular problem by finding code examples or answers online, at one of the many excellent technical communities that are available free. But, lacking a more robust understanding of the software frameworks employed or the possible side effects of certain coding practices, the alleged solution passes routine QA but fails in the long run. And even if the software works, performance suffers due to poor underlying architectural design.

What's to be done?

A large part of the solution is **real training** that delivers architectural understanding and expert guidance regarding your application domain. In-depth coverage, comprehensive hands-on labs and personal facilitation help you develop the thorough, practical competence that saves you time and boosts quality of work product. That is why we at Software Skills Training focus on all three of those elements of excellent IT training in every course we deliver, whether the course is delivered live face-to-face, remote-live, or via facilitated on-demand streaming.

Best Regards,

Phineas Longstaff
Software Skills Training, Inc.

Phineas Longstaff, the CEdO (Chief Education Officer) at Software Skills Training.com, has worked in the field of technical education for almost 40 years as a technical trainer, programmer, course developer, media developer, manager, executive, entrepreneur, writer, editor, and author. He welcomes your comments, criticism and input. Please contact him at Phineas.Longstaff@software-skills-training.com.