THINK LIKE A PROGRAMMER

An Introduction to Creative Problem Solving

by V. Anton Spraul

Do you struggle to write programs, even though you think you understand programming languages? Are you able to read through a chapter in a programming book, nodding your head the whole way, but find yourself unable to apply what you've read to your own programs? Are you able to comprehend a program example you've studied, even to the point where you could explain to someone else what each piece of the code is doing, yet you feel your brain seize up when faced with a programming task and a blank screen in your text editor?

You're not alone. I have taught programming for over 15 years, and most of my students would have fit this description at some point in their instruction. We will call the missing skill *problem solving*, the ability to take a given problem description and write an original program to solve it. Not all programming requires extensive problem solving. If you're just making minor modifications to an existing program, debugging, or adding testing code, the

San Francisco

Problem solving is hard. It's true that a few people make it look easy—the programming world's equivalent of a gifted athlete, like Michael Jordan. For these select few, high-level ideas are effortlessly translated into source code. To make a Java metaphor, it's as if their brains execute Java. Not being a natural isn't fatal to becoming a programmer—if it were, the world would have few programmers. Yet I've seen too many worthy learners...