tasks, including user-interface development, graphics, communications, database management, and networking, are readily available. Some libraries are in the public domain, some are open source, and some are sold commercially.

- ***Adopt a sensible set of coding conventions.*** A coding convention is a style rule that a programmer has decided to adopt even though it's not enforced by the language. Well-chosen conventions help make programs more uniform, easier to read, and easier to modify. Conventions are important when using any programming language, but especially so with C. As noted above, C's highly flexible nature makes it possible for programmers to write code that is all but unreadable. The programming examples in this book follow one set of conventions, but there are other, equally valid, conventions in use. (We'll discuss some of the alternatives from time to time.) Which set you use is less important than adopting *some* conventions and sticking to them.
- ***Avoid "tricks" and overly complex code.*** C encourages programming tricks. There are usually several ways to accomplish a given task in C: programmers are often tempted to choose the method that's most concise. Don't get carried away; the shortest solution is often the hardest to comprehend. In this book, I'll illustrate a style that's reasonably concise but still understandable.
- ***Stick to the standard.*** Most C compilers provide language features and library functions that aren't part of the C89 or C99 standards. For portability, it's best to avoid using nonstandard features and libraries unless they're absolutely necessary.

Q & A

Q: What is this Q&A section anyway?

A: Glad you asked. The Q&A section, which appears at the end of each chapter, serves several purposes.

The primary purpose of Q&A is to tackle questions that are frequently asked by students learning C. Readers can participate in a dialogue (more or less) with the author, much the same as if they were attending one of my C classes.

Another purpose of Q&A is to provide additional information about topics covered in the chapter. Readers of this book will likely have widely varying backgrounds. Some will be experienced in other programming languages, whereas others will be learning to program for the first time. Readers with experience in a variety of languages may be satisfied with a brief explanation and a couple of examples, but readers with less experience may need more. The bottom line: If you find the coverage of a topic to be sketchy, check Q&A for more details.

On occasion, Q&A will discuss common differences among C compilers. For example, we'll cover some frequently used (but nonstandard) features that are provided by particular compilers.