- ***Flexibility.*** Although C was originally designed for systems programming, it has no inherent restrictions that limit it to this arena. C is now used for applications of all kinds, from embedded systems to commercial data processing. Moreover, C imposes very few restrictions on the use of its features; operations that would be illegal in other languages are often permitted in C. For example, C allows a character to be added to an integer value (or, for that matter, a floating-point number). This flexibility can make programming easier, although it may allow some bugs to slip through.
- ***Standard library.*** One of C's great strengths is its standard library, which contains hundreds of functions for input/output, string handling, storage allocation, and other useful operations.
- ***Integration with UNIX.*** C is particularly powerful in combination with UNIX (including the popular variant known as Linux). In fact, some UNIX tools assume that the user knows C.

Weaknesses

C's weaknesses arise from the same source as many of its strengths: C's closeness to the machine. Here are a few of C's most notorious problems:

- ***C programs can be error-prone.*** C's flexibility makes it an error-prone language. Programming mistakes that would be caught in many other languages can't be detected by a C compiler. In this respect, C is a lot like assembly language, where most errors aren't detected until the program is run. To make matters worse, C contains a number of pitfalls for the unwary. In later chapters, we'll see how an extra semicolon can create an infinite loop or a missing & symbol can cause a program crash.
- ***C programs can be difficult to understand.*** Although C is a small language by most measures, it has a number of features that aren't found in all programming languages (and that consequently are often misunderstood). These features can be combined in a great variety of ways, many of which—although obvious to the original author of a program—can be hard for others to understand. Another problem is the terse nature of C programs. C was designed at a time when interactive communication with computers was tedious at best. As a result, C was purposefully kept terse to minimize the time required to enter and edit programs. C's flexibility can also be a negative factor; programmers who are too clever for their own good can make programs almost impossible to understand.
- ***C programs can be difficult to modify.*** Large programs written in C can be hard to change if they haven't been designed with maintenance in mind. Modern programming languages usually provide features such as classes and packages that support the division of a large program into more manageable pieces. C, unfortunately, lacks such features.