

# THE C PROGRAMMING LANGUAGE

Second Edition

**Brian W. Kernighan • Dennis M. Ritchie**

AT&T Bell Laboratories  
Murray Hill, New Jersey



PRENTICE HALL, Englewood Cliffs, New Jersey 07632

Let us begin with a quick introduction to C. Our aim is to show the essential elements of the language in real programs, but without getting bogged down in details, rules, and exceptions. At this point, we are not trying to be complete or even precise (save that the examples are meant to be correct). We want to get you as quickly as possible to the point where you can write useful programs, and to do that we have to concentrate on the basics: variables and constants, arithmetic, control flow, functions, and the rudiments of input and output. We are intentionally leaving out of this chapter features of C that are important for writing bigger programs. These include pointers, structures, most of C's rich set of operators, several control-flow statements, and the standard library.

This approach has its drawbacks. Most notable is that the complete story on any particular language feature is not found here, and the tutorial, by being brief, may also be misleading. And because the examples do not use the full power of C, they are not as concise and elegant as they might be. We have tried to minimize these effects, but be warned. Another drawback is that later chapters will necessarily repeat some of this chapter. We hope that the repetition will help you more than it annoys.

In any case, experienced programmers should be able to extrapolate from the material in this chapter to their own programming needs. Beginners should supplement it by writing small, similar programs of their own. Both groups can use it as a framework on which to hang the more detailed descriptions that begin in Chapter 2.

## **1.1 Getting Started**

The only way to learn a new programming language is by writing programs in it. The first program to write is the same for all languages:

```
Print the words  
hello, world
```