7 Basic Types

Make no mistake about it: Computers process numbers—
not symbols. We measure our understanding (and control)
by the extent to which we can arithmetize an activity.

So far, we've used only two of C's ***basic*** (built-in) ***types:*** `int` and `float`. (We've also seen `_Bool`, which is a basic type in C99.) This chapter describes the rest of the basic types and discusses important issues about types in general. Section 7.1 reveals the full range of integer types, which include long integers, short integers, and unsigned integers. Section 7.2 introduces the `double` and `long double` types, which provide a larger range of values and greater precision than `float`. Section 7.3 covers the `char` type, which we'll need in order to work with character data. Section 7.4 tackles the thorny topic of converting a value of one type to an equivalent value of another. Section 7.5 shows how to use `typedef` to define new type names. Finally, Section 7.6 describes the `sizeof` operator, which measures the amount of storage required for a type.

7.1 Integer Types

C supports two fundamentally different kinds of numeric types: integer types and floating types. Values of an ***integer type*** are whole numbers, while values of a floating type can have a fractional part as well. The integer types, in turn, are divided into two categories: signed and unsigned.

Signed and Unsigned Integers

The leftmost bit of a ***signed*** integer (known as the ***sign bit***) is 0 if the number is positive or zero, 1 if it's negative. Thus, the largest 16-bit integer has the binary representation