- In some C compilers, the ranges for `int` and `long int` are the same. That's because the compiler (usually a 32-bit model) can more efficiently handle `long` values than it can handle smaller `int` values. It's merely technical junk; don't memorize it or let it otherwise ruin your day.
- Negative numbers — why bother? Sometimes, you need them, but most of the time you don't. See the next section.
- The `char` variable type can also be used as a type of integer, though it has an extremely small range. These variables are used mostly to store single characters (or strings), which is discussed somewhere else. (Give me a second to look.) Oh, it's in Chapter 10.

Signed or unsigned, or "Would you like a minus sign with that, Sir?"

I have this thing against negative numbers. They're good only when you play Hearts. Even so, that's justification because you may someday write a program that plays Hearts on the computer, in which case you will be in dire need of negative numbers (because you can write the program so that you always win).

When you declare a numeric variable in C, you have two choices: *signed* and *unsigned*. Everything is signed unless you specifically type **unsigned** before the variable type:

```
unsigned int shoot_the_moon = 26;
```

A signed type of number means that a variable can hold a negative value. The standard `int` variable can hold values from –32,768 up to 32,767. That's half negative numbers, from –32,786 to –1, and then half positive numbers, from 0 up to 32,767. (Zero is considered positive in some cults.)

An unsigned number means that the variable holds only positive values. This unsigned number moves the number range all up to the positive side — no negatives (the C language equivalent of Prozac). Your typical `unsigned int` has a range from 0 to 65,535. Negative numbers aren't allowed.

The `int` variable `elephants` holds the value 40,000. Try *that* with a signed `int`! Ha!

```
unsigned int elephants 40000;
```

Table 9-2 illustrates the differences between the variable types as far as the values they can hold are concerned.