

As an alternative, we could use `typedef` to make `Suit` a type name:

```
typedef enum {CLUBS, DIAMONDS, HEARTS, SPADES} Suit;
Suit s1, s2;
```

In C89, using `typedef` to name an enumeration is an excellent way to create a Boolean type:

```
typedef enum {FALSE, TRUE} Bool;
```

C99 has a built-in Boolean type, of course, so there's no need for a C99 programmer to define a `Bool` type in this way.

Enumerations as Integers

Behind the scenes, C treats enumeration variables and constants as integers. By default, the compiler assigns the integers 0, 1, 2, ... to the constants in a particular enumeration. In our `suit` enumeration, for example, `CLUBS`, `DIAMONDS`, `HEARTS`, and `SPADES` represent 0, 1, 2, and 3, respectively.

We're free to choose different values for enumeration constants if we like. Let's say that we want `CLUBS`, `DIAMONDS`, `HEARTS`, and `SPADES` to stand for 1, 2, 3, and 4. We can specify these numbers when declaring the enumeration:

```
enum suit {CLUBS = 1, DIAMONDS = 2, HEARTS = 3, SPADES = 4};
```

The values of enumeration constants may be arbitrary integers, listed in no particular order:

```
enum dept {RESEARCH = 20, PRODUCTION = 10, SALES = 25};
```

It's even legal for two or more enumeration constants to have the same value.

When no value is specified for an enumeration constant, its value is one greater than the value of the previous constant. (The first enumeration constant has the value 0 by default.) In the following enumeration, `BLACK` has the value 0, `LT_GRAY` is 7, `DK_GRAY` is 8, and `WHITE` is 15:

```
enum EGA_colors {BLACK, LT_GRAY = 7, DK_GRAY, WHITE = 15};
```

Since enumeration values are nothing but thinly disguised integers, C allows us to mix them with ordinary integers:

```
int i;
enum {CLUBS, DIAMONDS, HEARTS, SPADES} s;

i = DIAMONDS;    /* i is now 1          */
s = 0;           /* s is now 0 (CLUBS)      */
s++;            /* s is now 1 (DIAMONDS)  */
i = s + 2;       /* i is now 3             */
```

The compiler treats `s` as a variable of some integer type; `CLUBS`, `DIAMONDS`, `HEARTS`, and `SPADES` are just names for the integers 0, 1, 2, and 3.