ENUMERATION TYPES

An **enumeration type** is a type whose values are defined by a list of named values of type int. This programmer-defined type is very much like a list of declared constants. Enumeration types can be handy for defining a list of identifiers to use as the case labels in a switch statement.

When defining an enumeration type, you can use any int values and can define any number of named values. For example, the following enumeration type defines a named value for the length of each month:

```
enum MonthLength { JAN_LENGTH = 31, FEB_LENGTH = 28,
   MAR_LENGTH = 31, APR_LENGTH = 30, MAY_LENGTH = 31,
   JUN_LENGTH = 30, JUL_LENGTH = 31, AUG_LENGTH = 31,
   SEP_LENGTH = 30, OCT_LENGTH = 31, NOV_LENGTH = 30,
   DEC_LENGTH = 31 };
```

As this example shows, two or more named constants in an enumeration type can receive the same int value.

If you do not specify any numeric values, the identifiers in an enumeration type definition are assigned consecutive values beginning with `0`. For example, the type definition:

```
enum Direction { NORTH = 0, SOUTH = 1, EAST = 2,
                 WEST = 3 };
```

is equivalent to:

```
enum Direction { NORTH, SOUTH, EAST, WEST };
```

The form that does not explicitly list the **`int`** values is normally used when you just want a list of names and do not care about what values they have. You can also specify the integer equivalents of the named values:

```
enum MyEnum { ONE = 7, TWO, THREE, FOUR = -3, FIVE };
```

then ONE takes the value 17; TWO takes the next **`int`** value, 18; THREE takes the next value, 19; FOUR takes -3; and FIVE takes the next value, -2. In short, the default for the first enumeration constant is 0. The rest increase by 1 unless you set one or more of the enumeration constants.

Although the constants in an enumeration type are given as **`int`** values and can be used as integers in many contexts, remember that an enumeration type is a separate type and treat it as a type different from the type int. Use enumeration types as labels and avoid doing arithmetic with variables of an enumerations type.