Chapter 8 introduced several new data types, such as float and char. We discussed the range of each type and the format specifiers used to print each type using printf(). Next, you explored the concept of arrays, focusing on the relationship between char arrays and C strings. Along the way, you discovered the #define statement, C’s text substitution mechanism. In Chapters 7, 8 and 9, you learned a lot about pointers.

This chapter will show you how to use existing C types as building blocks to design your own customized data structures. You’ll also learn how to dynamically allocate memory for those structures as you need it.

Bundling Data

There will be times when your programs will want to bundle together, or associate, related data. For example, suppose you are writing a program to organize your DVD collection. Imagine the type of information you will want to access for each DVD. At the very least, you’ll want to keep track of the movie’s title. You might also want to rate each DVD on a scale from one to ten. Finally, let’s add in a comment field you can use to describe your feelings about the movie or perhaps note to whom you loaned this particular movie.

In the next few sections, we’ll look at two separate approaches to a basic DVD tracking program. Each approach will revolve around a different set of data structures. One will make use of arrays (Model A) and the other a set of custom designed data structures (Model B).
Model A: Three Arrays

One way to model your DVD collection is with a separate array for each DVD’s attributes:

```c
#define kMaxDVDs             5000
#define kMaxTitleLength      256
#define kMaxCommentLength    256

char    rating[ kMaxDVDs ];
char    title[ kMaxDVDs ][ kMaxTitleLength ];
char    comment[ kMaxDVDs ][ kMaxCommentLength ];
```

This code fragment uses three `#define` s. `kMaxDVDs` defines the maximum number of DVDs this program will track. `kMaxTitleLength` defines the maximum number of characters in a DVD title. `kMaxCommentLength` defines the maximum number of characters in the DVD comment array.

Next, `rating` is an array of 5,000 chars, one char per DVD. Each of the chars in this array will hold a number from 1 to 10, the rating you’ve assigned to a particular DVD. This line of code assigns a value of 8 to DVD 37:

```c
rating[ 37 ] = 8;    /* A pretty good DVD */
```

The arrays `title` and `comment` are each known as *multidimensional arrays*. A normal array, like `rating`, is declared using a single dimension. The statement of

```c
float   myArray[ 5 ];
```

declares a normal (one-dimensional) array containing five floats, namely:

```c
myArray[ 0 ]
myArray[ 1 ]
myArray[ 2 ]
myArray[ 3 ]
myArray[ 4 ]
```

This statement

```c
float   myArray[ 3 ][ 5 ];
```

declares a two-dimensional array containing 15 floats (3 * 5 = 15), namely:

```c
myArray[0][0]
myArray[0][1]
myArray[0][2]
myArray[0][3]
myArray[0][4]
myArray[1][0]
myArray[1][1]
myArray[1][2]
myArray[2][0]
```