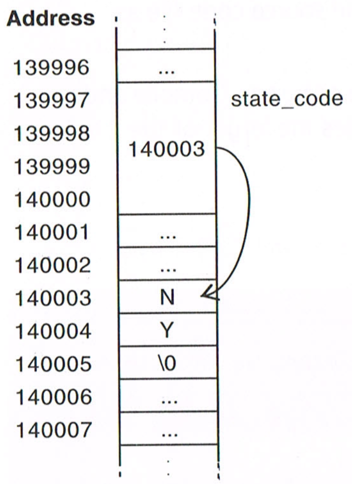
ALL CHARACTER ARRAYS USE A POINTER



F I G U R E 10-4
The name of a character array is actually a pointer constant.

The first example of a pointer in this chapter was a pointer to a character array. Let's look at that figure again (see Figure 10-4).

When you declare a character array, the name you give the array is actually a ***pointer constant.*** A pointer constant is like a ***pointer variable,*** except you cannot change what it points to. In Figure 10-4, the **state_code** pointer constant is initialized by the compiler to point to whatever location is assigned to the first character of the character array.

Note

If you were wondering why C++ requires you to use the **strcpy** *function to assign a string to a character array, here is why. A statement like the one below would try to assign a string of characters to a pointer constant.*

```
state_code = "NY"; // CAN'T DO THIS!
```

USING SUBSCRIPT NOTATION

C++ allows you to access any character in a character array individually using a method called ***subscript notation.*** Subscript notation looks like the syntax you use when declaring a character array. Consider the program in Figure 10-5. The character array **my_word** is declared to be five characters long and assigned the word *book.*

```
#include <iostream.h> // necessary for cout command

int main()
{
  char my_word[5] = "book";
  cout << my_word << '\n';
  return 0;
}
```

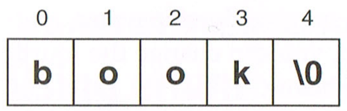
F I G U R E 10-5
In this program, the character array **my_word** is initialized with the word *book.*

Recall from Chapter 6 that the characters of a character array are stored together in memory. What you may not know, however, is that the characters of the array are numbered, beginning with zero (see Figure 10-6). So a five-character array is numbered 0 through 4.

Using subscript notation, you can change the first character in the array using a statement like the one below.

```
my_word[0] = 't';
```

The first character is changed to a 't' because the first character is at position 0. You may wonder why the people who created C++ made the subscript of the first character 0 rather than 1. You might find it helpful to think of the number in the brackets as the number of places you must move over from the first character in the array.



F I G U R E 10-6
The characters of an array are numbered, beginning with zero.

The first character in the array is where the pointer points. To access the fourth character, however, you must move three times. An exercise will give you a little practice with subscript notation.

EXERCISE 10-4 SUBSCRIPT NOTATION

1. Enter the program below. Save the source code as *WORDPLAY.CPP.*

```
#include <iostream.h> // necessary for cout command

int main()
{
  char my_word[5] = "book";

  cout << my_word << '\n';

  my_word[3] = 't';
  cout << my_word << '\n';
  return 0;
}
```

2. Compile and run the program to see how subscript notation changed one character in the word.
3. Add the following statements to the bottom of the program.

```
  my_word[0] = 's';
  cout << my_word << '\n';
```

4. Run the program again to see the effect of the statements you added.
5. Save the source code and leave it open for the next exercise.

PITFALLS

Be careful when accessing characters in an array individually. The compiler will not prevent you from going beyond the length of your array. You could end up changing other data in memory instead of your array. If the null terminator of your string is changed to another character, the compiler will allow it and errors will result.

Subscript notation can be used without knowledge of pointers. However, since you are becoming familiar with pointers, you would probably be interested in seeing how to change a character in the array without subscript notation.

USING THE * OPERATOR IN CHARACTER ARRAYS

You already know that **my_word** is a pointer to the first character in the array. Because characters occupy one byte of memory, it makes sense that the second