The `std` *Namespace*

The `std::` before `cout` is required when we use names that we've brought into the program by the preprocessing directive `#include <iostream>`. The notation `std::cout` specifies that we are using a name, in this case `cout`, that belongs to namespace `std`. The names `cin` (the standard input stream) and `cerr` (the standard error stream)—introduced in Chapter 1—also belong to namespace `std`. Namespaces are an advanced C++ feature that we discuss in depth in Chapter 23, Other Topics. For now, you should simply remember to include `std::` before each mention of `cout`, `cin` and `cerr` in a program. This can be cumbersome—the next example introduces `using` declarations and the `using` directive, which will enable you to omit `std::` before each use of a name in the `std` namespace.

The Stream Insertion Operator and Escape Sequences

In the context of an output statement, the `<<` operator is referred to as the **stream insertion operator**. When this program executes, the value to the operator's right, the right **operand**, is inserted in the output stream. Notice that the operator points in the direction of where the data goes. A string literal's characters *normally* print exactly as they appear between the double quotes. However, the characters `\n` are *not* printed on the screen (Fig. 2.1). The backslash (\) is called an **escape character**. It indicates that a "special" character is to be output. When a backslash is encountered in a string of characters, the next character is combined with the backslash to form an **escape sequence**. The escape sequence `\n` means **newline**. It causes the **cursor** (i.e., the current screen-position indicator) to move to the beginning of the next line on the screen. Some common escape sequences are listed in Fig. 2.2.

Escape sequence	Description
`\n`	Newline. Position the screen cursor to the beginning of the next line.
`\t`	Horizontal tab. Move the screen cursor to the next tab stop.
`\r`	Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line.
`\a`	Alert. Sound the system bell.
`\\`	Backslash. Used to print a backslash character.
`\'`	Single quote. Used to print a single quote character.
`\"`	Double quote. Used to print a double quote character.

Fig. 2.2 | Escape sequences.

The `return` *Statement*

Line 10

```
    return 0; // indicate that program ended successfully
```

is one of several means we'll use to **exit a function**. When the **`return` statement** is used at the end of `main`, as shown here, the value `0` indicates that the program has *terminated successfully*. The right brace, `}`, (line 11) indicates the end of function `main`. According to the