Nope! All the `cases` are done. What's left is the *default,* which supposedly handles everything else — including the `X`:

```
default:
```

and the only statement:

```
printf("I don't know that key.\n");
```

The `switch-case` structure is done.

- The thing in `switch`'s parentheses (*`choice`*) must work out to either a character value or an integer value. Most programmers put a character or integer variable there. You can also put a C language statement or function in the parentheses, as long as it works out to a character value or an integer value when it's done.
- The `case` line ends with a colon, not a semicolon. The statements belonging to `case` aren't enclosed in curly braces.
- The last statement belonging to a group of `case` statements is usually `break`. This statement tells the computer to skip over the rest of the `switch` structure and keep running the program.
- If you forget the `break`, the rest of the `switch` structure keeps running. That may not be what you want.
- The computer matches each item in the `case` statement with the choice that `switch` is making. If there's a match, the statements belonging to that `case` are executed; otherwise, they're skipped.
- It's possible for a `case` to lack any statements. In that case, a match simply "falls through" to the next `case` statement.
- The keyword `case` must be followed by a constant value — either a number or a character. For example:

  ```
  case 56:          /* item 56 chosen */
  ```

  or

  ```
  case 'L':         /* L key pressed */
  ```

  You cannot stick a variable there. It just doesn't work. You may want to. You may even e-mail me, asking whether you can, but you can't. Give up now.
- Most C manuals refer to the command as `switch`, and `case` is just another keyword. I use `switch-case` as a unit because it helps me remember that the second word is `case` and not something else.
- You don't need a `default` to end the structure. If you leave it off and none of the `case`'s items matches, nothing happens.