To summarize this chapter: When using TeX for straight copy, you type the copy as on an ordinary typewriter, except that you need to be careful about quotation marks, the number 1, and various kinds of hyphens/dashes. TeX will automatically take care of other niceties like ligatures and kerning.

(Are you sure you should be reading this paragraph? The "dangerous bend" sign here is meant to warn you about material that ought to be skipped on first reading. And maybe also on second reading. The reader-beware paragraphs sometimes refer to concepts that aren't explained until later chapters.)

If your keyboard does not contain a left-quote symbol, you can type `\lq`, followed by a space if the next character is a letter, or followed by a `\` if the next character is a space. Similarly, `\rq` yields a right-quote character. Is that clear?

`\lq\lq␣I␣understand.\rq\rq␣`

In case you need to type quotes within quotes, for example a single quote followed by a double quote, you can't simply type ''' because TeX will interpret this as "' (namely, double quote followed by single quote). If you have already read Chapter 5, you might expect that the solution will be to use grouping—namely, to type something like {'}''. But it turns out that this doesn't produce the desired result, because there is usually less space following a single right quote than there is following a double right quote: What you get is '", which is indeed a single quote followed by a double quote (if you look at it closely enough), but it looks almost like three equally spaced single quotes. On the other hand, you certainly won't want to type '␣'', because that space is much too large—it's just as large as the space between words—and TeX might even start a new line at such a space when making up a paragraph! The solution is to type '\thinspace'', which produces '" as desired.

▶ **EXERCISE 2.4**
OK, now you know how to produce "' and '"; how do you get " ' and ' "?

▶ **EXERCISE 2.5**
Why do you think the author introduced the control sequence `\thinspace` to solve the adjacent-quotes problem, instead of recommending the trickier construction '$\,$'' (which also works)?

> *In modern Wit all printed Trash, is*
> *Set off with num'rous* Breaks——*and* Dashes——
> — JONATHAN SWIFT, *On Poetry: A Rapsody* (1733)

> *Some compositors still object to work*
> *in offices where type-composing machines are introduced.*
> — WILLIAM STANLEY JEVONS, *Political Economy* (1878)