Is X negated and then squared, or is it squared and then negated? Fortran reference manuals seldom treat such fine points in detail; this may be a hard question to answer without running a test program. As a matter of fact the ANSI standard for Fortran calls for the latter interpretation (fortunate in this case) — the variable X is squared and then negated — but the line should still be rewritten as:

```
TERM = -TERM * X**2 / DENOM
```

The first form invites misunderstanding on the part of the reader, if not the compiler. Unless reader and compiler both understand the writer, the program is not communicating properly.

Parenthesize to avoid ambiguity.

Variable names can also be either safe or dangerous:

```
8 NO5S = NO5S + 1
```

Now was that "N, letter O, five, S," or "N, zero, five, S," or even "NOSS"? The possibilities for error are numerous. Would you trust someone else to type corrections for this program? Mixtures of similar characters (letter O and digit 0, letter I and digit 1, etc.) are unsafe, as are long identifiers that differ only at the end. Use XPOS and YPOS, not POSITIONX and POSITIONY. When abbreviating, always keep first letters, favor "pronounceable" forms (XPOS, not XPSTN), and above all be consistent.

Similar identifiers are dangerous in general. One program contains the improbable sequence

```
N = K
N = K**2
NNN = K**3
```

It is only when, much further down, we read

```
WRITE(6,60)N,NN,NNN, ...
```

that the typographical error in the second line becomes clear. A better choice of names here is N, NSQ, NCUBE. Try to choose names that differ widely; typos and misspellings are less likely to be disguised. Of course, choose names that mean something as well, so the intent of the code is clearer. (We will discuss this more in Chapter 8.)

Choose variable names that won't be confused.

We have discussed arithmetic expressions quite a bit, but conditional expressions are at least as important in writing programs. In either PL/I or Fortran, conditional expressions nearly always involve at least one IF statement, which controls whether or not another statement is executed, on the basis of some condition. PL/I