```
      DIMENSION SALESM(17)
      DO 20 I = 1, 17
         SALESM(I) = MOD(I,2) + 1
   20 CONTINUE
```

We are more concerned here with the mnemonic values chosen: even-numbered salesmen have an odd number, while odd-numbered salesmen have an even number. This is backwards. It is certainly no more difficult to let "1" be the code for odd-numbered salesmen, and "2" the code for even. (We leave it to the reader to decide whether using a floating point array to hold integer values is an appropriate data representation.)

Programs should not only cope with incorrect input, but also encourage users to make fewer errors by being easy to use. Consider this input fragment:

```
      READ(1,4) XIHP(I),R(I),FORCE(I),NREV(I)
    4 FORMAT(F5.1,F3.1,F3.0,I2)
```

If you are an occasional user of this program, will you be able to remember, a week from now, that the XIHP field is five columns wide, R and FORCE are three, and NREV is two? Not likely. There would be less to remember if the author had used a uniform format, like

```
    4 FORMAT(3F5.0, I5)
```

in which all fields have the same width. This also leaves room for future growth — whatever NREV is, it may someday be bigger than 99, which is all the I2 format permits. I2 is not only non-uniform, but restrictive.

You might even consider doing this:

```
      READ(1,4) XIHP(I), R(I), FORCE(I), REV
    4 FORMAT(4F5.0)
      NREV(I) = REV
```

Now we do not need to right-justify the integer NREV — we enter it with a decimal point like everything else, and convert it internally. (As it turns out, the only place in the program where NREV is used is in a long product where all the other factors are floating point. It might as well have been declared floating point anyway.)

The difference between F3.1 and F3.0, by the way, lies in the interpretation of an input number that does not contain a decimal point. If the number contains a decimal point, as it should for reliability, the explicit point overrides the decimal point position in the FORMAT statement. Floating point input specifications should thus be restricted to the form Fn.0. (We have quietly made this change in most of our examples so far.) Omitting decimal points to make fields smaller is penny-wise, pound-foolish.

Use uniform input formats.
