POINTS TO PONDER

8.1 Programming in a standard or stereotyped way is often a useful way to avoid error. For example, in Fortran, identifiers that begin with I, J, K, L, M, or N are integer by default, and all others are floating point. This convention is widely used. Sometimes, however, to avoid straining for meaningful identifiers, it seems easier to declare variables explicitly, overriding the default. From the standpoints of error potential and reader comprehension, is this good practice or bad? You might consider this excerpt in your deliberations:

```
C A SORTING PROGRAM
      ...
      INTEGER X, Y
      DIMENSION X(25), Y(25)
      ...
            IF ( X(I) .LE. X(J) ) GO TO 20
               TEMP = X(I)
               X(I) = X(J)
               X(J) = TEMP
               TEMP = Y(I)
               Y(I) = Y(J)
               Y(J) = TEMP
```

8.2 Fortran continuation lines are often left behind when statements are moved within a program. What practices can you think of, in writing multi-line statements, that would reduce the likelihood of your making this mistake (or at least ensure that the compiler will spot your error)? Look back over the Fortran programs in this book.

8.3 Comment on these comments:

```
      DO 65 L=1,9999
C        GENERATE RANDOM NUMBER
41    CALL RANDU(IX,IY,YFL)
C        SET NEW VALUE OF IX TO VALUE OF IY
      IX=IY
C        COMPUTE SAMPLE WHICH IS TO RECEIVE BACTERIA
      N=YFL*100.0 +1.0
C        CHECK TO SEE IF N IS 101
      IF(N-101) 40,41,40
C        CHECK TO SEE IF SAMPLE ALREADY CONTAINS BACTERIA
40    IF(IT(N) )21,20,21
C        INCREMENT NUMBER OF SAMPLES CONTAINING BACTERIA BY ONE
20    ICT=ICT+1
C        INCREMENT NUMBER OF BACTERIA IN SAMPLE BY ONE
21    IT(N)=IT(N)+1
C        CHECK TO SEE IF 50 OF THE SAMPLES CONTAIN BACTERIA
      IF(ICT-50)65,33,33
65    CONTINUE
```