5.3 Fortran ignores most blanks in program statements, but treats most blanks as zeros in input data. Thus

```
      READ(5,10) N
   10 FORMAT(I5)
      IF (N .EQ. 1 024) WRITE(6,20) N
   20 FORMAT(1X, I5)
```

will cause the input number

```
1 024
```

to be stored as 10024 by the READ statement, but compared to 1024 in the IF. Can you think of any benefit to be gained from this inconsistency? How can you avoid any trouble it might cause?

5.4 The following PL/I program counts words and sentences and computes averages. Debug it, then revise it along the lines suggested in this chapter.

```
            DECLARE  TEXT  CHARACTER(200) VARYING,
                     CHAR  CHARACTER(1),
                    (SENT,  /* NO. OF SENTENCES */
                     WORDS, /* NO. OF WORDS */
                     LETTERS)  FIXED ;

START:      /* INITIALIZE AND READ TEXT */
            SENT, WORDS, LETTERS = 0 ;
            GET LIST( TEXT ) ;

            /* EXAMINE TEXT FOR WORDS AND SENTENCES, AND
               COUNT LETTERS IN THE PROCESS */

            DO I = 1 BY 1 TO LENGTH( TEXT ) ;
               CHAR = SUBSTR( TEXT, I, 1 ) ;
               IF CHAR='.' THEN
                  DO ;
                     /* A PERIOD ENDS A WORD AND A SENT. */
                     WORDS = WORDS + 1 ;
                     SENT = SENT + 1 ;
                  END ;
               ELSE IF CHAR=' ' THEN WORDS = WORDS + 1 ;
               ELSE  LETTERS = LETTERS + 1 ;

            /* NOTE THE ASSUMPTION THAT A CHARACTER IS
               CONSIDERED TO BE A LETTER IF IT IS NOT
               A PERIOD OR A BLANK. */

            END ;

            /* PRINT RESULTS */
            PUT SKIP EDIT( 'SENTENCES =', SENT,
                           'WORDS/SENTENCE =', WORDS/SENT,
                           'LETTERS/WORDS =', LETTERS/WORDS )
                          ( X(10), A, F(4) ) ;
            GO TO START ;
```