```
IF WING < 0.6 * LENGTH THEN
   CORR = 1.0 - 0.037;
ELSE
   CORR = 1.0 + 0.045;

IF LENGTH >= 80 THEN
   WEIGHT1 = (CORR+0.122) * WEIGHT;
ELSE IF LENGTH > 60 THEN
   WEIGHT1 = (CORR+0.105) * WEIGHT;
ELSE IF LENGTH > 50 THEN
   WEIGHT1 = (CORR+0.09) * WEIGHT;
ELSE IF LENGTH >= 30 THEN
   WEIGHT1 = (CORR+0.08) * WEIGHT;
```

Both versions use a nest of IF-ELSE's, but ours uses it in a special way: we never use an IF immediately after a THEN, but only after an ELSE. The result is much easier to understand, because we know that exactly one case is done (if LENGTH exceeds 30), and it is clear how we get to it: it is the first condition satisfied as we read down the list of ELSE IF's. After one of these has been done, execution resumes after the entire statement. Of course this is just a CASE statement.

The construction THEN IF is often a warning that trouble looms ahead.

```
IF QTY > 10 THEN                                    /*A*/
   IF QTY > 200 THEN                                /*B*/
      IF QTY >= 500 THEN BILL_A = BILL_A + 1.00;    /*C*/
                    ELSE BILL_A = BILL_A + .50;     /*C*/
                ELSE;                               /*B*/
           ELSE BILL_A = .00;                       /*A*/
```

Those letters down the right hand side are designed to help you figure out what is going on, but as usual, no amount of commenting can rescue bad code. The sequence of THEN-IF's requires you to maintain a mental pushdown stack of what tests were made, so that at the appropriate point you can pop them until you determine the corresponding action (if you can still remember). You might time yourself as you determine what this code does when QTY equals 350. How about 150?

Since at most one of a set of actions is ever called for here, what we really want is some form of CASE statement. Changing the order in which the decisions are made leads to a clearer version:

```
IF QTY >= 500 THEN
   BILL_A = BILL_A + 1.00;
ELSE IF QTY > 200 THEN
   BILL_A = BILL_A + 0.50;
ELSE IF QTY <= 10 THEN
   BILL_A = 0.0;
```

Now all we need do is read down the list of tests until we find one that is met, read across to the corresponding action, and continue after the last ELSE.

In Fortran, this could be rendered as

```
IF (QTY .GE. 500.0) BILLA = BILLA + 1.0
IF (QTY .LT. 500.0 .AND. QTY .GT. 200.0) BILLA = BILLA + 0.5
IF (QTY .LE. 10.0) BILLA = 0.0
```

which is best if the tests are mutually exclusive and if the relations and actions are