The WHILE statement, which specifies a loop with an arbitrary termination condition (tested at the top), is not available in Fortran. This means that loops in Fortran are often contorted into DO loops, which makes them hard to understand and prone to errors. Alternatively, loops are written with IF's and GOTO's, which conceal the structure of code, again making it hard to understand and prone to errors. Newer languages have better control-flow constructs, like PL/I's DO-WHILE. But the DO-WHILE is seldom used, and PL/I programmers often write much as their Fortran colleagues do.

Use DO *and* DO-WHILE *to emphasize the presence of loops.*

Things get more complicated when several fundamental structures are intertwined and built with spare parts instead of being spelled out explicitly, as in this excerpt from a procedure that computes bowling scores.

```
      Y=0; L=1; FRM=1;
CYCLE:   IF X(L) = 10 THEN  STRK: DO;
                                  Y=Y+10+X(L+1)+X(L+2);
                                  L=L+1;
                                  GO TO NEXT;
                                  END STRK;
         IF X(L) + X(L+1) = 10 THEN  SPR: DO;
                                          Y=Y+10+X(L+1);
                                          L=L+2;
                                          GO TO NEXT;
                                          END SPR;
                               ELSE  REG: DO;
                                          Y=Y+X(L)+X(L+1);
                                          L=L+2;
                                          GO TO NEXT;
                                          END REG;
NEXT: IF FRM=10 THEN RETURN(Y);
    FRM=FRM+1;   GO TO CYCLE;
```

There are actually two structures here, both built with IF's and GOTO's instead of with the higher-level facilities provided by PL/I. The outer part is an indexed loop, represented by

```
          FRM=1;
CYCLE:
   ...
NEXT: IF FRM=10 THEN RETURN(Y);
    FRM=FRM+1;   GO TO CYCLE;
```

and the interior is a three-way decision: strike, spare, or regular frame.

Rewritten with explicit control structures, it becomes much clearer. Bowlers will appreciate the correction of the computation for a spare; non-bowlers may be less interested.