(B+C) just happens to equal the monthly payment M (within one-half cent)? The program takes an extra trip around the loop, recomputes B and C, and informs the user that "THERE WILL BE A LAST PAYMENT OF: 0.00". This is graceless.

We can patch the code to read

```
IF B+C < 0.005 THEN GOTO L10; /* FINAL PAYMENT ALREADY MADE */
IF B+C < M THEN GOTO L20;
```

but it would be better to reorganize it completely, so that it reads from top to bottom instead of branching about.

There is also an instance here of a familiar error. Notice that there are two exits from the DO loop, one from the side and one from the bottom, that arrive at the same place. This should always arouse suspicion. The program is designed to exit from the side when the loan is paid off with sixty payments or fewer, and from the bottom when it is not. What happens if it does exit from the bottom? The final payment is B+C, but although B has been recomputed to be the correct remaining balance, C is not recomputed before being used — it is the interest charge left over from the previous payment. Clearly the interest charge should be either zero, or recomputed from the new B, depending on whether the final payment is made immediately or after another month.

In either case, this is an error. (It must be coincidental that it is in the bank's favor.) Our version avoids this problem with the safer DO-WHILE construction. As we repair the code we can correct the illegal F(6,21) format item in the first PUT statement, add a SKIP so the headings are placed on a new line instead of being tacked onto the end of whatever message was printed previously, and eliminate the unnecessary variable B and the poorly-named labels.