```
    LOW = 1;
    HIGH = LIMIT;                    /* LIMIT = TABLE SIZE */
  SEARCH_AGAIN:
    MID = (LOW + HIGH)/ 2;
    IF HIGH <= LOW THEN
      CALL ARG_NOT_FOUND;
    ELSE
      IF SEARCH_ARG = TABLE(MID) THEN
        CALL PROCESS_TABLE_FUNCTION;
      ELSE
        DO;
          IF SEARCH_ARG > TABLE(MID) THEN
            LOW = MID + 1;
          ELSE
            HIGH = MID - 1;
          GO TO SEARCH_AGAIN;
        END;
```

We will be talking more about binary search shortly, but for now, observe that if the table contains only one entry, then HIGH and LOW are both 1, and so the routine decides that the desired value, SEARCH_ARG, is not in the table without ever looking at either value or table! The problem, of course, is that the "<=" should be "<". (As an exercise, find the other cases for which this code fails.)

Take care to branch the right way on equality.

Here is another instance where branching the wrong way on equality results in a small but real error. The code computes a table of monthly balances and interest charges for a given principal amount, interest rate, and monthly payment.

```
DECLARE (A,R,M,B,C,P)   FIXED DECIMAL (13,4);

L10:GET LIST (A,R,M);
    PUT EDIT ('THE AMOUNT IS',A)(A(13),F(10,2))
             ('   THE INTEREST RATE IS',R)(A(23),F(6,21))
             ('   THE MONTHLY PAYMENT IS',M)(A(25),F(8,2));
    IF M<=A*R/1200 THEN GO TO L30;
    PUT SKIP(3)EDIT
('        MONTH      BALANCE   CHARGE       PAID ON PRINCIPAL')(A);
    PUT SKIP;
    B=A;
DO I=1 TO 60;
  C=B*R/1200;
  IF B+C<M THEN GO TO L20;
  P=M-C;  B=B-P;
  PUT SKIP EDIT (I,B,C,P)(F(13),3 F(13,2));
END;
L20:PUT SKIP(2)EDIT ('THERE WILL BE A LAST PAYMENT OF:',B+C)
                                                (A(35),F(8,2));
    GO TO L10;
L30:PUT SKIP(2)EDIT ('UNACCEPTABLE MONTHLY PAYMENT')(A);
    GO TO L10;
```

What happens if the amounts are such that the balance due plus the interest charge