ALGORITHM 45
INTEREST

PETER Z. INGERMAN
University of Pennsylvania, Philadelphia, Pa.

```
procedure monpay (i, B, L, t, k, m, tol, goof)
comment This procedure calculates the periodic payment
  necessary to retire a loan when the interest rate on the loan
  varies (possibly from period to period) as a function of the as-
  yet-unpaid principal.
    The formal parameters are: i, array identifier for the vector
  of interest rates; −B, array identifier for the minimum amounts
  at which the corresponding i applies; −L, the amount to be
  borrowed; −t, the number of periods for which the loan is to
  be taken out; −k, the number of different interest rates (and
  upper limit for vectors i and B); −m, the desired periodic pay-
  ment; −tol, the allowable deviation of m from some ideal;
  and goof, the error exit to use if convergence fails. The only
  output parameter is m. For further discussion, see Comm.
  ACM 3 (Oct. 1960), 542;
begin array h, S [1:k, 1:t], M, X [1:k];
integer array T, a, b [1:k];
integer p, q, r, sa, sb, I, ib, mb, nb;
comment This section sets up the procedure;
for p := 1 step 1 until k do
begin for q := 1 step 1 until t do
    begin h[p,q] := i[p]↑q;
          S[p,q] := (h[p,q] − 1)/(i[p] − 1) end;
    if p = 1 then X[p] := 0 else X[p] := B[p] × (i[p−1] − i[p]);
    M[p] := L × (h[p,t]/S[p,t]) end;
sa := sb := ib := mb := 0; nb := t;
for p := 1 step 1 until k do
begin a[p] := entier (B[p+1]/M[p+1] + 0.5) − sa;
      sa := sa + a[p];
      T[p] := b[p] := entier (B[p+1]/M[p] − 0.5) − sb;
      sb := sb + b[p];
      if b[p] > mb then
      begin ib := p; nb := nb − mb; mb := bp end
      else nb := nb − b[p] end;
T[ib] := nb;
I := 1;
for p := 1 step 1 until k do
    I := I × (a[p] − b[p] + 1);
comment Having counted the number of possible iterations
  and established a set of trial values for the T[n]'s, a trial m is
  found;
D := 1; E := F := 0;
newm: for p := 1 step 1 until k do
      begin D := D × h[p,Tp];
            u := 1;
            if p ≠ 1 then for q := 1 step 1 until p − 1
              do u := u × h[q,Tq];
            E := E + S[p,Tp] × u;
            v := 0;
            if p ≠ 1 then for r := 1 step 1 until p
              do v := v + X[r];
            F := F + u × v end;
      m := (L × D + F)/E;
      comment Now find out whether m is good enough
      q := 1; F := D := 0;
      for p := 1 step 1 until t do
      begin get F: F := (D + m − E)/(1 + i[q]);
            if B[q+1] ≧ F then D := F else q := q + 1;
            if D ≠ F go to get F end;
            if abs (D − L) ≦ tol then go to exit;
      comment If not within tolerance, adjust T[n]'s and try
        again;
      p := 0;
      redo: p := p + 1;
          if p ≠ ib then
          begin if T[p] ≧ a[p] then
                begin T[ib] := T[ib] + T[p] − b[p]
                      T[p] := b[p] end end
          else begin
              T[p] := T[p] + 1;
              T[ib] := T[ib] − 1;
              p := k end;
          if p = k then I := I − 1 else go to redo;
          go to if I > 0 then newm else goof;
exit: end monpay;
```

CERTIFICATION OF ALGORITHM 45
INTEREST [Peter Z. Ingerman, *Comm. ACM* Apr. 1961 and Oct. 1960]

CARL B. WRIGHT
Dartmouth College, Hanover, N. H.

INTEREST was translated into Dartmouth College Computation Center's "Self Contained ALGOL Processor" for the Royal-McBee LGP-30. When using SCALP, memory capacity is severely limited and thus it was necessary to run this program in two blocks. Block I ended with the computation of I, and Block II started with the "*newm*" loop. After making the changes listed below, test problems using up to three interest rates and up to 18 time periods were used with the following results:

Loan	*Periods*	*Interest Rates*	*Payments*	*Final Balance**	*Tolerance*
$100.00	1	0.05	$105.00	$0.00	$0.25
1800.00	10	0.03	211.01	0.05	4.50
875.65	8	0.08 to 500.00 0.05 over 500.00	139.78	−1.49	2.19
14750.00	18	0.06 to 5000.00 0.05 to 10,000.00 0.04 over 10,000.00	1201.70	10.56	36.88

* Hand calculation.

It is noted that in each case the final balance is within the prescribed tolerance (0.0025 of the loan).

In the following corrections bracketed subscripts replace ordinary subscripts and exponentiation is represented by ↑ rather than superscript.

The following corrections should be made in the Note on Interest in the October, 1960, issue of *Comm. ACM*:

1. Definition of $B[n]$: Replace "*minimum*" by "*maximum*". Replace "$j[n]$" by "$j[n-1]$".
2. Define $B[k+1] \equiv L$.
3. Definition of $K[n]$: Replace "$B[n]$" by "$B[n+1]$".

The following corrections were found necessary in the procedure:

1. The upper limit of the vector B is $k+1$, not k. It is not necessary to change the upper limit of the I-vector. (See correction 4 below.)

2. D, E, F, u, v were not declared and must be declared as **real**.

3. In the **array** declaration replace "$M[1:k]$" by "$M[1:k+1]$".

4. As j approaches 0, i approaches 1 and lim $(h/S) = 1/t$. Thus for $j[k+1] = 0$, $i[k+1] = 1$, and $M[k+1] = L/t$. Thus after

$M[p] := L \times (h[p,t]/S[p,t])$ **end**;

insert

$M[k+1] := L/t$; $B[k+1] := L$;

5. In the conditional statement following computation of $b[p]$, replace "$>$" by "$\geqq$".

6. In same conditional statement, next line, "$mb := bp$" should read "$mb := b[p]$".

7. $D := 1$; $E := F := 0$;
newm: for $p := 1$ **step** 1 **until** k **do**

should be changed to

newm: $D := 1$; $E := F := 0$;
for $p := 1$ **step** 1 **until** k **do**

8. **begin** *get F*: $F := (D+m-E)/(1+i[q])$;
if $B[q+1] \geqq F$ **then** $D := F$ **else** $q := q + 1$;
if $D \neq F$ **go to** *get F* **end**;

should be changed to read as follows:

begin *get F*: $F := (D+m)/i[q]$;
if $B[q+1] \geqq F$ **then** $D := F$ **else**
begin if $q < k$ **then** $q := q + 1$ **else** $D := F$ **end**;
if $D \neq F$ **then go to** *get F* **end**;

Note that the "then" in the last line was omitted from the original procedure.

9. In the "*redo*" loop insert a semicolon after the statement
$T[ib] := T[ib] + T[p] - b[p]$;

10. In the "*redo*" loop, next line, omit the second "*end*".

11. In the "*redo*" loop,

$p := k$ **end**;

should be changed to

$p := k$ **end end**;