ALGORITHM 18
RATIONAL INTERPOLATION BY CONTINUED FRACTIONS

R. W. Floyd
Armour Research Foundation, Chicago, Illinois

```
comment  This procedure fits to m given points (xi , yi) a con-
           tinued fraction in the form
    a1+(x-x1)/(a2+(x-x2)/(a3+(x-x3)/···(x-xm-1)/am))···))
    It also simplifies the continued fraction to a rational function
    (N0+N1x+···+Ndeg x^deg)/(D0+D1x+···+Ddeg x^deg),
    where deg is at most m ÷ 2;
procedure  confr(m,x,y,a,N,D);
real array  x,y,a,N,D;  integer m;
begin real  aa, xx, T;  integer i, j, k;  real array P,Q[0 : m ÷ 2]
    switch sw := sw1, sw2;
    for j := 1 step 1 until m do
    begin aa := y[j];  xx := x[j];
        for i := 1 step 1 until j-1 do
        aa  := (xx-x[i])/(aa-a[i]);  a[j]  :=  aa
    end;
    k := 1;  P[0] := 1;  Q[0] := a[1];
    mult : for j := 1 step 1 until m ÷ 2 do P[j] := Q[j] := 0;
    for i := 2 step 1 until m do
    begin for j := i ÷ 2 step -1 until 1 do
        begin T := a[i] × Q[j] - x[i-1] × P[j] + P[j-1];
            P[j] := Q[j]; Q[j] := T
        end;  T := a[i] × Q[0] - x[i-1] × P[0];
        P[0] := Q[0];  Q[0] := T
    end;  go  to  sw[k];
    sw1 : for j := 0 step 1 until m ÷ 2 do N[j] := Q[j];
            k := 2;  P[0] := 0; Q[0] := 1;  go to mult;
    sw2 : for j := 0 step 1 until m ÷ 2 do D[j] := Q[j]
  end  procedure
```

CERTIFICATION OF ALGORITHM 18
RATIONAL INTERPOLATION BY CONTINUED FRACTIONS

[R. W. Floyd, *Comm. ACM.*, Sept. 1960]

Henry C. Thacher, Jr.*
Reactor Engineering Div., Argonne National Lab., Argonne, Ill.

* Work supported by the U. S. Atomic Energy Commission

The body of procedure *confr* was tested with the Algol translator system written for the LGP-30 computer by the Dartmouth College Computer Center. No syntactical errors were found in the procedure body, except for a missing semicolon after the array delcaration. The translated algorithm gave satisfactory results when tested on values of $(4x + 1)/(x + 4)$ at any three of the points $x = 1, 2, 3, 4$. When all four points were used, a division overflow occurred in the statement **for** i := 1 **step** 1 **until** $j-1$ **do** aa := $(xx - x[i])/(aa-a[i])$; which forms the reciprocal differences. An overflow of this type will occur whenever $y[j]$ is approximated to high accuracy by one of the continued fractions based only on the points $x[i]$, $i = 1, 2, \cdots, k$ with k less than j. Unless $i = j-1$, the difficulty may be overcome by setting aa equal to the largest real representable in the computer whenever division overflow would occur. When $i = j-1$, the difficulty is irretrievable, and the data points must be reordered.