ALGORITHM 95
GENERATION OF PARTITIONS IN PART-COUNT FORM
FRANK STOCKMAL
System Development Corp., Santa Monica, Calif.

procedure partgen(c,N,K,G); integer N,K; integer array c;
Boolean G;

comment This procedure operates on a given partition of the
positive integer N into parts \( \leq K \), to produce a consequent
partition if one exists. Each partition is represented by the
integers \( c[1] \) thru \( c[K] \), where \( c[j] \) is the number of parts of the
partition equal to the integer \( j \). If entry is made with \( G = \text{false} \),
procedure ignores the input array \( c \), sets \( G = \text{true} \), and pro-
duces the first partition of \( N \) ones. Upon each successive entry
with \( G = \text{true} \), a consequent partition is stored in \( c[1] \) thru \( c[K] \).
For \( N = KX \), the final partition is \( c[K] = X \). For \( N = KX+r \),
\( 1 \leq r \leq K-1 \), final partition is \( c[K] = X \), \( c[r] = 1 \). When entry
is made with array \( c = \text{final partition} \), \( c \) is left unchanged and \( G \)
is reset to \( \text{false} \);

begin integer a,i,j;
    if \( \neg G \) then go to first;
    j := 2;
    a := c[1];
    test: if \( a < j \) then go to B;
    c[j] := 1 + c[j];
    c[1] := a - j;
    zero: for i := 2 step 1 until j - 1
        do c[i] := 0;
        go to EXIT;
B: if \( j = K \) then go to last;
    a := a + j \times c[j];
    j := j + 1;
    go to test;
first: G := \text{true};
    c[1] := N;
    j := K + 1;
    go to zero;
last: G := \text{false};
EXIT: end partgen