ALGORITHM 245
TREESORT 3 [M1]
Robert W. Floyd (Reed. 22 June 1964 and 17 Aug. 1964)

procedure TREESORT 3 $(M, n)$;
value $n$; array $M$; integer $n$;

comment TREESORT 3 is a major revision of TREESORT
[R. W. Floyd, Alg. 113, Comm. ACM 6 (Aug. 1962), 434] sug-
spected by HEAPSORT [J. W. J. Williams, Alg. 232, Comm.
ACM 7 (June 1964), 347] from which it differs in being an in-place
sort. It is shorter and probably faster, requiring fewer compar-
isons and only one division. It sorts the array $M[1:n]$, requiring
no more than $2 \times (2^p-2) \times (p-1)$, or approximately $2 \times
n \times (\log(n)-1)$ comparisons and half as many exchanges in the
worst case to sort $n = 2^p - 1$ items. The algorithm is
most easily followed if $M$ is thought of as a tree, with $M[j+2]$
the father of $M[j]$ for $1 < j \leq n$;

begin
\begin{verbatim}
procedure exchange $(x, y)$; real $x, y$;
    begin real $t$; $t := x$; $x := y$; $y := t$
    end exchange;

procedure siftup $(i, n)$; value $i, n$; integer $i, n$;
    comment $M[i]$ is moved upward in the subtree of $M[1:n]$ of
    which it is the root;
    begin real copy; integer $j$;
    copy := $M[i]$;
    loop: $j := 2 \times i$;
    if $j \leq n$ then
    begin if $j < n$ then
        begin if $M[j+1] > M[j]$ then $j := j + 1$ end;
        if $M[j] > copy$ then
        begin $M[i] := M[j]$; $i := j$; go to loop end
    end;
    $M[i] := copy$
    end siftup;

integer $i$;
for $i := n+2$ step $-1$ until $2$ do siftup $(i, n)$;
for $i := n+2$ step $-1$ until $2$ do
begin siftup $(1, i)$;
    comment $M[j+2] \geq M[j]$ for $1 < j \leq i$;
    exchange $(M[1], M[i])$;
    comment $M[1:n]$ is fully sorted;
end
end TREESORT 3
\end{verbatim}

ALGORITHM 246
GRAYCODE [Z]
J. Boothroyd* (Reed. 18 Nov. 1963)
English Electric-Leo Computers, Kidsgrove, Stoke-on-
Trent, England
* Now at University of Tasmania, Hobart, Tasmania, Aust.

procedure graycode $(a)$ dimension: $(n)$ parity: $(s)$; value $n, s$;
Boolean array $a$; integer $n$; Boolean $s$;

comment elements of the Boolean array $a[1:n]$ may together be
considered as representing a logical vector value in the Gray
cyclic binary code. [See e.g. Phister, M., Jr., Logical Design of
procedure changes one element of the array to form the next
value in ascending sequence if the parity parameter $s = true$
or in descending sequence if $s = false$. The procedure
may also be applied to the classic “rings-o-seven” puzzle [see
K. E. Iverson, A Programming Language, p. 63, Ex. 1.5];
begin integer $i, j$; $j := n + 1$;
for $i := n$ step $-1$ until $1$ do if $a[i]$ then begin $s := \neg s$;
    $j := i$ end;
if $s$ then $a[1] := \neg a[1]$ else if $j < n$ then $a[j+1] := \neg a[j+1]$;
else $a[n] := \neg a[n]$
end graycode

ALGORITHM 247
RADICAL-INVERSE QUASI-RANDOM POINT
SEQUENCE [G5]
J. H. Halton and G. B. Smith (Reed. 24 Jan. 1964 and
21 July 1964)
Brookhaven National Laboratory, Upton, N. Y., and
University of Colorado, Boulder, Colo.

procedure QRPSH $(K, N, P, Q, R, E)$;
integer $K, N$; real array $P, Q$; integer array $R$; real $E$;

comment This procedure computes a sequence of $N$ quasi-
random points lying in the $K$-dimensional unit hypercube
given by $0 < x_i < 1$, $i = 1, 2, \cdots, K$. The $i$th component of the
$m$th point is stored in $Q[m,i]$. The sequence is initiated by a
“zero-th point” stored in $P$, and each component sequence is
iteratively generated with parameter $R[i]$. $E$ is a positive error-
parameter. $K$, $N$, $E$, and the $P[i]$ and $R[i]$ for $i = 1, 2, \cdots, K$,
are to be given.

The sequence is discussed by J. H. Halton in Num. Math. 2
(1960), 84-90. If any integer $n$ is written in radix-$R$ notation as
$n = n_0 \cdots n_{m-1}$, $0 = n_0 + n_1 R + n_2 R^2 + \cdots + n_{m-1} R^{m-1}$,
and reflected in the radical point, we obtain the $R$-inverse func-
tion of $n$, lying between 0 and 1,
\[
\phi_R(n) = n_0 n_1 n_2 \cdots n_{m-1} = n_0 R^{m-1} + n_1 R^{m-2} + \cdots + n_{m-1} R^0
\]
\[
= n_0 R^{m-1} + \cdots + n_{m-1} R^0.
\]
The problem solved by this algorithm is that of giving a com-
plete procedure for the addition of $R^i$, in any radix $R$, to a frac-
tion, with downward “carry”.

If $P[i] = \phi_R(s)$, as will almost always be the case in practice,
with $s$ a known integer, then $Q[m,i] = \phi_R(s+m)$. For quasi-
randomness (uniform limiting density), the integers $R[i]$ must
be mutually prime.

For exact numbers, $E$ would be infinitesimal positive. In prac-
tice, round-off errors would then cause the “carry” to be in-
correctly placed, in two circumstances. Suppose that the stored
number representing $\phi_E(n)$ is actually $\phi_R(n) + \Delta$. (a) If $|\Delta|$ \[
\geq R^{-m-1}, \text{we see that the results of the algorithm become un-}
\]