ALGORITHM 63
PARTITION
C. A. R. HOARE
Elliott Brothers Ltd., Borehamwood, Hertfordshire, Eng.

procedure partition (A,M,N,I,J); value M,N;
array A; integer M,N,I,J;
comment I and J are output variables, and A is the array (with
subscript bounds M:N) which is operated upon by this procedure.
Partition takes the value X of a random element of the array A,
and rearranges the values of the elements of the array in such a
way that there exist integers I and J with the following properties:
M ≤ J < I ≤ N provided M < N
A[R] ≤ X for M ≤ R ≤ J
A[R] = X for J < R < I
A[R] ≥ X for I ≤ R ≤ N

The procedure uses an integer procedure random (M,N) which
chooses equiprobably a random integer F between M and N, and
also a procedure exchange, which exchanges the values of its two
parameters;
begin real X; integer F;
F := random (M,N); X := A[F];
I := M; J := N;
up: for I := I step 1 until N do
if X < A[I] then go to down;
I := N;
end up;
down: for J := J step -1 until M do
if A[J]<X then go to change;
J := M;
end down;
change: if I < J then begin exchange (A[I], A[J]);
I := I + 1; J := J - 1;
go to up
end change;
else if I < F then begin exchange (A[I], A[F]);
I := I + 1
end;
else if F < J then begin exchange (A[F], A[J]);
J := J - 1
end;
end partition

CERTIFICATION OF ALGORITHMS 63, 64 AND 65,
PARTITION, QUICKSORT, AND FIND [Comm. ACM, July 1961]
B. RANDELL AND L. J. RUSSELL
The English Electric Company Ltd., Whetstone, England

Algorithms 63, 64, and 65 have been tested using the Pegasus
ALGOL 60 Compiler developed at the De Havilland Aircraft
Company Ltd., Hatfield, England.
No changes were necessary to Algorithms 63 and 64 (Partition
and Quicksort) which worked satisfactorily. However, the
comment that Quicksort will sort an array without the need for any
extra storage space is incorrect, as space is needed for the organiza-
tion of the sequence of recursive procedure activations, or, if
implemented without using recursive procedures, for storing
information which records the progress of the partitioning and
sorting.
A misprint ('if' for 'else' on the line starting 'else if J ≤ K then
...') was corrected in Algorithm 65 (FIND), but it was found that
in certain cases the sequence of recursive activations of Find
would not terminate successfully. Since Partition produces as
output two integers J and I such that elements of the array
A[M:N] which lie between A[J] and A[I] are in the positions that
they will occupy when the sorting of the array is completed, Find
should cease to make further recursive activations of itself if K
fulfills the condition J < K < I.
Therefore the conditional statement in the body of Find was
changed to read

if K ≤ J then find (A,M,J,K)
else if I ≤ K then find (A,J,N,K)

With this change the procedure worked satisfactorily.