Finding the Median

The median of the folowing 7 items

16 12 99 95 18 87 10

is 18 as half the items are ≤ 18 and half are ≥ 18.
The item 18 is the middle item in size and so is the middle item when sorted.

We can find the median of a sequence by sorting the sequence and then getting the item at position $\frac{n+1}{2}$ but sorting has best performance O(n*log n) while a 'find median' algorithm developed by C.A.R. Hoare has performance O(n).

Sorting the above sequence we get

10 12 16 18 87 95 99

The middle (4th) item is 18 and so 18 is the median.

O(n) Algorithm for Finding Median

By adapting the Partition procedure, Hoare developed an O(n) algorithm (he called it Find) that would find the median. In fact, Hoare's 'Find' algorithm is more general, it finds the K^{th} smallest item and so with K=$\frac{n+1}{2}$, the Find algorithm gets the median.
Hoare's algorithm is based on repeated use of Partition.

Assume we want to find the 4^{th} smallest item is the above sequence,

16 12 99 95 18 87 10
L R

By continuing to partition about the 4th item (i.e. using the 4th item -- 95 as the pivot) we get

16 12 10 87 18 95 | 99
L R

As the left split has length ≥ 4, the 4th smallest item must be in the left section. We again partition the left section about the 4th item (now = 87) to get

16 12 10 18 87 || 95 | 99
L R

The size of the current left section is still ≥ 4 and so the 4th smallest item must be in this section and so we partition again to get

16 12 10 18 ||| 87 || 95 | 99
L R

We continue partitioning until L ≥ R. At this final stage the 4th smallest item will be at position 4. The array values have been moved around by partition but the array is not sorted; we have the 4th smallest item is in its proper position if the array was sorted.

The Eiffel procedure for Find

Assume the sequence is stored in the array attribute, A, and that we have a procedure

Partition(L0,R0 : INTEGER; P : G)

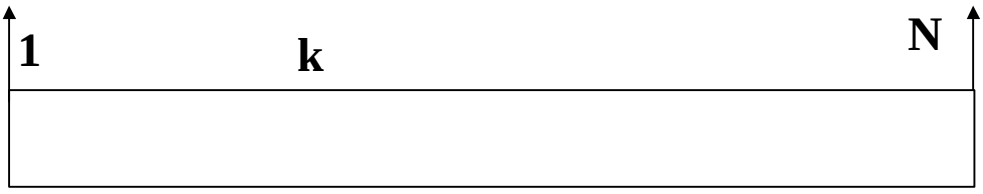
which partitions an array segment A[L0..R0] about a pivot P. Assume also that we have class attributes
L, R : INTEGER, which keep track of partition splits.

Hoare's Find procedure is:

```
find (k, left, right : INTEGER) is
--   find the kth smallest item
--   in the array segment A[left..right]
     local
          i, j : INTEGER
          p : G
     do
          from
               i := left
               j : = right
          until
               i >= j
          loop
               p := A.item(k)
               partition(i,j,p)          -- L:=i; R:=j
                                         -- finishes with R < L
               if R < k then
               -- kth smallest in right split
                    i := L
               end
               if k < L then
               -- kth smallest in left split
                    j := R
               end
          end
     end -- find
```

For reference,we repeat the procedure for partition.

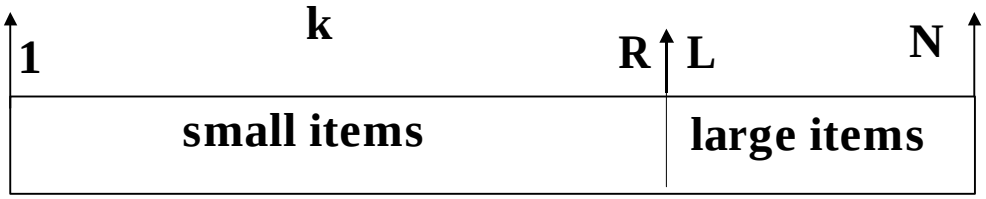
```
Partition (L0,R0 : INTEGER;  P : G) is
    do
        from
            L := L0
            R := R0
        until
            L > R
        loop
            Left_Scan (P)
            Right_Scan(P)
            if L <= R then
                exchange(L,R)
                L := L+1
                R := R-1
            end
        end
    end -- Partition

    Left_Scan (P : G) is
        do
            from
            until
                A.item(L) >= P
            loop
                L := L+1
            end
        end -- Left_Scan

    Right_Scan (P : G) is
        do
            from
            until
                A.item(R) <= P
            loop
                R := R-1
            end
        end -- Right_Scan
```



Move small values left

Move large values right

After Partition about A.item(k)



A.item(k) has new value

Example: