(F.3) Given a sequence of n (n $>$ 2) real numbers in an array, A[1], A[2], A[3], ..., A[n], we wish to find two numbers A[i] and A[j], where i $<$ j, such that A[i] $\leq$ A[j] and their distance is the largest. That is,

$$j - i = \max\{v - u \mid 1 \leq u < v \leq n \text{ and } A[u] \leq A[v]\}$$

If no such pair, report -$\infty$. Please design an $\mathcal{O}(n \cdot lgn)$ algorithm to solve this problem. You can use any method , including divide and conquer, greedy or dynamic programming.

**The following should be noted about the algorithm below (in addition to the statements in the problem):**

1. Function SortAndTrackIndices(A, I) returns array A sorted in non-decreasing order ($a_1 \leq a_2 \leq a_3 \leq ... \leq a_n$) as well as an array, $I$, which maps each $a_i$ to its original position, $I_i$.

**Algorithm 1** Greedy solution to the problem above. Make sure to take note of the assumptions above.

```
1:  function MaxDist(A, n)
2:      A, I ← SortAndTrackIndices(A)
3:      i_max ← j_max ← I[n]
4:      i ← j ← I[n]
5:
6:      for k from n - 1 to 1 do
7:          if j < I[k] then
8:              i ← I[k]
9:              j ← I[k]
10:         else
11:             if i > I[k] then
12:                 i ← I[k]
13:             end if
14:         end if
15:         if j − i > j_max − i_max then
16:             i_max ← i
17:             j_max ← j
18:         end if
19:     end for
20:
21:     if i_max equals j_max then
22:         return -∞
23:     else
24:         return i and j
25:     end if
26:
27: end function
```