1. (10 pts) Describe an algorithm for finding the second largest integer in a sequence of distinct integers. Give a big-O estimate of the number of comparisons used by your algorithm.

procedure secondmax ($a_1, a_2, \ldots, a_n$: distinct integers)

if $a_1 > a_2$ then (max := a_1 AND secondmax := a_2)
else (max := a_2 AND secondmax := a_1)

One comparison

$i := 3$

while $i \leq n$

if $a_i >$ max then (secondmax := max AND max := a_i)
elseif $a_i >$ secondmax then secondmax := a_i

at most 2 comparisons t each iteration in while

$i := i + 1$

return secondmax

Note that the # of comparisons used is linear in n. That is, at most a constant # of comparisons is used at each iteration and there are $\approx n$ (actually $n-2$) iterations. Thus, running time $O(n)$

Note Some people sorted the array and returned the second largest element. This is fine, but then the running time (say, using merge sort) is $O(n \log n)$