CS2223
HW#3 SOLUTIONS

1. (*a*) Guess (& verify in *n*-1 comparisons) that the list is sorted. If it isn't, sort it. Return the next to the last element.

```
sorted=1;
for (i=0;i<n-1;i++)
        if (A[i]>A[i+1]) sorted=0;    /* executed n-1 times*/
if (!sorted) InsertionSort(A);
return A[n-2];
```

(*b*) The algorithm must be able to identify the *n*-2 "losers" (elements which are not among the two largest), and hence identify the two "winners". If the *n*-2 losers weren't identified (didn't each lose a comparison), then one of them could be changed to ∞ and the algorithm would still identify it as a loser (all the comparisons returned the same result). If, having identified the two winners, they were never compared to each other, then the algorithm couldn't identify the smaller of the two (if the input reversed them, then the algorithm would persist in identifying the now incorrect second largest).

2. (*a*) $\lfloor \lg n \rfloor$

(*b*) If the input is sorted, in increasing or decreasing order, the height is *n*-1. Note that these are not the only worst-case instances.