CS2223
HW#3

DUE: Monday, November 17

1. (10 points) Consider the computational model in which the benchmark operation is a pairwise comparison (the algorithm can select any two objects from the set and ask which is larger). In class we showed that according to this model, the problem of finding the largest element of a set of n distinct elements must use at least n-1 comparisons. Consider the problem of finding the second largest element of a set of n elements.
(*a*) Describe an algorithm to identify the second largest element such that the best case performance of the algorithm uses n-1 pairwise comparisons.
(*b*) Show that according to this computational model, **any** algorithm to identify the second largest element of the set must also be able to identify the largest. That is, after performing its sequence of comparisons, enough information must have been gleaned to permit the identification of the largest **and** the second largest elements, and to know which is larger.

2. (14 points) Consider the data structure *search tree* (described on pp. 157→159 of our text) to implement the abstract data type *set*.
(*a*) What is the smallest possible height of a search tree of n nodes?
(*b*) What is the largest height possible for a search tree of n nodes? Describe an input permutation which yields this worst-case height.
(*c*) In order to estimate the average height (assuming all input permutations occur with equal probability) of a search tree of n nodes, program and test routines to make a search tree, to insert a node into a search tree, and to compute the height of a search tree.

- Grow random search trees of heights 2^3, 2^4, 2^5,… nodes.
- Compute the heights of your trees.
- See if you can find a constant c such that the heights are approximately $c\lg n$.