Homework#4 Divide and Conquer

Textbook:

6.35. Let $A[1..n]$ be a set of integers. Give an algorithm to reorder the elements in $A$ so that all negative integers are positioned to the left of all nonnegative integers. Your algorithm should run in time $\Theta(n)$.

6.2. Consider Algorithm SLOWMINMAX which is obtained from Algorithm MINMAX by replacing the test

\[
\text{if high} - \text{low} = 1
\]

by the test

\[
\text{if high} = \text{low}
\]

and making some other changes in the algorithm accordingly. Thus, in Algorithm SLOWMINMAX, the recursion is halted when the size of the input array is 1. Count the number of comparisons required by this algorithm to find the minimum and maximum of an array $A[1..n]$, where $n$ is a power of 2. Explain why the number of comparisons in this algorithm is greater than that in Algorithm MINMAX. (Hint: In this case, the initial condition is $C(1) = 0$).

6.52. Give a divide-and-conquer algorithm to find the second largest element in an array of $n$ numbers. Derive the time complexity of your algorithm.

Counting inversions. An inversion in an array $a[]$ is a pair of entries $a[i]$ and $a[j]$ such that $i<j$ but $a[i]>a[j]$. Given an array, design a linearithmic ($O(n\log n)$) algorithm to count the number of inversions.

Space complexity of Quicksort. Modify QUICKSORT to ensure its work space is $\Theta(\log n)$. 