Selection Problem and complexity.

The selection problem consists of finding the k-th largest element in an array. Write the following methods to solve the selection problem using a randomly-generated array of integers as input:

a) Sort the array using bubble sort, then return the k-th element of the sorted array
b) Sort the array using quicksort, then return the k-th element of the sorted array
c) Use a modification of quicksort to solve the problem. Observe that quicksort uses a pivot to partition the array into a subarray that has elements that are less than or equal to the pivot and a subarray that has elements that are larger than the pivot. By looking at the size of the first subarray, you can figure out if the k-th largest element is in the first or second subarray, and make a recursive call to search in that subarray. Thus you will need just one recursive call instead of the two quicksort makes.

Determine analytically the big O running time of each of the four algorithms.

Run all three methods and find their running times using various values of n (for example 100; 1,000; 10,000; 100,000; 1,000,000) and a value of k = n/2, for any n you use. Do your experimental result match your analytical results?

Write a report describing your work. We are particularly interested in your observations about the behavior of each algorithm as the size of the input data increases and also in the comparison of different algorithms for each input size. You may want to use graphs or plots to illustrate this. Also, feel free to run and report additional experiments for a more thorough analysis.