Net ID: wpicc001
Name: William Patrick Picca

Sources Used: Introduction to Algorithms, Third Edition - Chapter 2.1, Insertion Sort

Problem A:
Submission ID: 129949908

For part 1, Finding the K-th student, I approached the problem using insertion sort. In reading the problem and having to sort an inputted array of integers, I decided to use Insertion Sort as the sorting algorithm because it was the first sorting method I read about in the textbook. What it does is start with the first 2 elements of the entire array and compares the 2, switching positions if need be. It then expands the number of elements being checked by 1, comparing the newly comparable element to that of the now sorted element at the end of our sorted array. If the comparison is made and deems necessary, it will insert our new element before the last, and continue comparing to the remaining elements, repeating the entire process until the sorting is finished.
I approached the problem based on the individual functions that would be needed to perform the entire problems, followed by the actual sorting needed to be done, to be ended with the outputting the final results.
After having sorted the array of integers correctly, and merely writing the input statements for the remaining queries related to the problem, I had to write a nested forloop, where the outer loop is dependent on the number of queries, and the inner loop is dependent on the length of the array of integers.
Finally the inner loop is what obtains and outputs the answers to the designated queries, using a series of if statements. This is done by obtaining the largest number of the sorted array at the end of the list, then deducting in the element position of the array the desired query number. This in turn gives the ranked number as desired by the program.

Problem B:
Submission ID: 129954978

Problem B was done in nearly the exact same fashion as Problem A, allowing me to use nearly the exact same code I had. The only changes I had to make were in if statements within the nested forloop. I checked every element in the integer array to that of the queries in question. If a desired number is found, the element's numbered position within the array, then subtracted off of the array size, is then outputted to display the required ranked position of that number.

Problem C:

Submission ID: 129956019

Problem C used the exact same code as A and B except for the nested forloops as the end of the code. For this problem in particular, having to obtain the highest distinct scores requires setting up a counter to count the number of distinct scores having been iterated. Every time an element of the already sorted integer array is examined, it is compared to the last integer element examined. If it is a unique number, the counter is incremented. At the end of every iteration of the inner forloop, the counter is compared to the queried distinct number, outputting the current value of that element if applicable.