CS3350 Assignment 2 Due: Sunday, Sep.24 11:59 p.m.

You are given a Password.txt, which lists the most used 10k passwords in the world. Your goal is to sort the passwords in the file in an ascending order (characters are compared by the ASCII codes) by using various sorting algorithms and make a comparison of the algorithms.

**Your Tasks:**

1. Write a single assignment2\_yourLastname. cpp file that includes **A**. a main function to call the sorting algorithms to sort the Password.txt, **B**. the implementation of all the following sorting algorithms (some of them are already in Zybook and you can re-use the codes).
2. Bubble sort
3. Selection sort (Select the smallest in each iteration)
4. Insertion sort
5. Merge sort
6. Quick Sort

For quick sort, you must use the **median-of-three pivot selection** introduced in class: you need to sort the first, middle and last elements and then use the middle element as the pivot.

1. Set two counters for each sorting algorithm.
2. Text

   Description automatically generatedCounter 1 is used to record the **number of comparisons**. Whenever you have a comparison of two passwords during the sorting process, you need to increase one for this counter. Print the counter value after the sorting is done.
3. Counter 2 is used to record the **number of movements.** Whenever you move one password to a different location (a swap typically costs 3 movements), you need to increase one for this counter. Print the counter value after the sorting is done.

The output sample is displayed on the right. Your program will be outputting the number of comparisons and movements that replace “XXX” in this sample.

1. Write a report to explain how you implemented each of the sorting functions briefly and whether your calculations of the comparisons and movements match the printed results. Here is a sample explanation for a bubble sort implementation:

*“I followed the traditional bubble sort algorithm and ran 'n' passes to sort the array. In each pass, my function pushed the largest element to the end of the current unsorted array by comparing the adjacent values. If there are n unsorted elements in the array, there requires n-1 comparisons in this pass. Therefore, the number of comparisons calculated is calculated as 1+2+3+…+(n-1), which equals n(n-1)/2= 49,995,000. This matches the printed result. The number of movements should be a large number but less than the three times the number of comparison because each swap requires three movements, but not all the comparisons lead to a movement.”*

**Zip & Submit**

1. **Submit the Assignment2\_** **yourLastname.cpp** to the Assignment2 link on Canvas.
2. **Submit your report Assignment2\_** **yourLastname.doc to** the Assignment2 link on Canvas.

**Grading Rubric**

\_\_\_\_\_\_\_ (3) proper documentation/comments where appropriate

\_\_\_\_\_\_\_ (3) implementation of Bubble sort and explanation in report

\_\_\_\_\_\_\_ (3) implementation of Selection sort and explanation in report

\_\_\_\_\_\_\_ (3) implementation of Insertion sort and explanation in report

\_\_\_\_\_\_ (4) implementation of Merge sort and explanation in report

\_\_\_\_\_\_\_ (4) implementation of Quicksort and explanation in report

**The number of comparisons and number of movements will vary based on your implementation. You must correctly implement a sorting algorithm and explain the printed values in your report.**