Instructions

Assignment 5

For this assignment, you will complete searching/sorting tasks and efficiency analysis. No code is to be written for this assignment.

Submitting

Submit your answers in a PDF file called **assign5.pdf** in Sakai under Written Assignments. Use your preferred text editor to type your answers, and then save the file in PDF format.

You must upload your file, and then click the Submit button. If you don't click the Submit button we don't have your assignment. You'll receive an email for each submission. If you don't, submit it again, you have unlimited submissions. Don't submit at the last minute, the system might be overloaded with submissions and won't be able to register yours.

Problem 1

1. List the resulting array after each iteration of the **selection** sort algorithm. Indicate the number of letter-to-letter comparisons made for each iteration.
   Sort the following array of letters (sort into alphabetical order):

```
2. C Q S A X B T
```

3. List the resulting array after each iteration of the **insertion** sort algorithm. Indicate the number of letter-to-letter comparisons made for each iteration.
   Sort the following array of letters (sort into alphabetical order):

```
4. C Q S A X B T
```

Problem 2

For each problem given below, do the following:

1. Create an algorithm in pseudocode to solve the problem.
2. Identify the factors that would influence the running time of your algorithm. *For example, if your algorithm is to search an array the factor that influences the running time is the array size.* Assign names (such as n) to each factor.
3. Determine the number of operations in each step of the pseudocode. To do that, identify the operations that must be counted. You need not count every statement separately. If a group of statements always executes together, treat the group as a single unit. If a method is called, and you do not know the running time of that method, count it as a single operation.
4. Count the operations performed by the algorithm. Express the count as a function of the factors you identified in Step 2.
5. Describe the best case scenario for the algorithm and derive the big O.

6. Describe the worst case scenario for the algorithm and derive the big O.

```
a. Determine if two arrays have no elements in common.

b. Counting the total number of characters that have a duplicate within a string. (i.
e. "gigi the gato" would result in 7 (g x 3 + i x 2 + t x 2)

c. Finding a row where every entry is 'x' in a 2-D array.
```