# CS 2401 Assignment #7

**Due Date:** Sunday, November 11, 2018 11:59PM (See the syllabus for late policy).

**Objective:** The goal of this assignment is to learn about empirical performance testing to compare the speed of different algorithms.

**Deliverables:** You are expected to submit a total of two files, one of which is java code `Search.java` and one MS Word document `Report.docx`. All the files must be submitted using Blackboard. Sometimes, docx files do not open properly on other computers. Make PDF of the docx file and submit the PDF too (that makes it a total of three files).

**Assignment:** Your goal is to determine an empirical estimate of the efficiency of two algorithms. To do this you need to be able to calculate the execution time of a method. This can be done by using a system method to query the current system time before and after executing the method; the time elapsed is the difference between the two. In Java, you can use the method call `System.currentTimeMillis()` to get the time in milliseconds as a long. If the execution times are very small you can also use `System.nanoTime()`. This method is not as accurate, but you can use it to get greater precision if you are consistently seeing millisecond times of 0.

**Main task:** Implement two search algorithms, linear and binary, that work on an array of doubles.

> **Java files:** `Search.java` should contain two search techniques: (1) a linear search to search in an array of doubles, (2) a binary search method to search in an ordered array.
> **Other requirements:** `Search.java` must have (and you must use it) a method that implements linear search and binary search algorithms. Each method must return the location of the element searched for in an array, or return -1 if the element is not in the array.
> **More requirements:** To get an accurate estimate of the time taken, you will need to generate a number of test cases and average the resulting times. Suppose first that we aim to estimate the time taken for a list of 10000 numbers. Here are the steps to perform:
>
> a. Generate a new array with 10000 elements, and initialize the elements to random values using Math.random().
> b. Sort the array using Arrays.sort(). Remember that binary search works on ordered arrays only.
> c. Select the value to search for by selecting a random index between 0 and 9999 and using that value (so you can be sure the value is in the array). Select the random index using Math.random().
> d. Run linear and binary search for the same element on the sorted array, and

record the elapsed time for each algorithm.

e. Repeat the above steps at least 30 times using a for loop and calculate the average time taken by each algorithm. This will be more accurate than doing the test only once.

Now that you are able to test the performance of the linear and binary search algorithms on arrays of 10000 elements, the next step is to test problems of several different sizes to see how the runtime changes as the problem size increases. Repeat the experiment described above with 30 samples for arrays with the following sizes: 10000, 40000, 160000, 640000, 1280000 (these number may vary based on the performance of your computer). Generate a plot in Excel showing the performance trend of the two methods, with the array size on the *x*-axis and the average runtime on the *y*-axis. Draw two lines, one for the linear search and the other for the binary search algorithm, on the same plot. You can output the results to the terminal and copy the values into Excel by hand. Copy the Excel plot in your MS Word report.

**Report:** In addition to your codes, submit a short report in a docx file (`Report.docx`) describing your experiments and observations. The MS Word document must contain the plots you have drawn using excel. The report should include: a description of the running environment (computer, processor, memory, OS, Java version), description of the experiments including the steps that you took to gather the data, the results (i.e., the Excel plots you generated), and a discussion of factors that could influence the accuracy of the results. Provide big O notation for each of the algorithms (1) linear search, and (2) binary search. Explain how your runtime plots relate to the big O notations.