Assignment 1
Searching in Arrays

Release Time	Due Date
January 22, 2017	**February 5, 2017**

Objectives

- Experience various techniques to search in sorted arrays
- Practice developing high-performance solutions
- Compare theoretical vs empirical complexities
- Compare iterative and recursive solutions of a problem

Problem Specification

Binary search is a common technique to search in a sorted array. Of course, one can implement binary search iteratively (i.e., using loops) or recursively (i.e., using recursive methods). In this assignment we will get experience in implementing and comparing their performances. We will also gain experience in extending binary search. Here we go.

Write a JAVA application to solve the following problem:

1) Generate n random floating-point numbers from 1 to m i.e., in the range [1, m)

   (Hint: http://stackoverflow.com/questions/20389890/generating-a-random-number-between-1-and-10-java);

2) Store the generated numbers in an array and sort the array. You may use Java's Arrays.sort utility method.

3) Exact-match query: Generate a random number x and search x in the sorted array using two implementations of binary search, first is an iterative implementation and the second is a recursive implementation.

4) Redo step 3 by extending the binary search to a ternary-search (i.e., "poke" the array at 1/3 and 2/3).

5) Redo step 3 by extending the binary search to a quad-search (i.e., "poke the array at ¼, ½ and ¾).

6) Range-Query: Generate two random numbers x and y and binary-search the sorted array to find the number of elements of the array between x and y (i.e., find out the count of the number of values in the range [x, y]).

7) Perform a theoretical complexity analysis of your design (i.e., count number of operations/instructions and space usage) and then express that using asymptotic notation as a function of the input size (Pause: what are the input and output sizes of your problems?)

8) Empirically measure the time complexity of your code.

9) Compare the complexities from steps 7) and 8). (Long pause: in order to compare, what all you will need to do? What all you must compare? We will discuss some alternatives in class, so stay tuned and this write-up may be modified accordingly at a later date.) Input m and r from the console (m is the upper bound on the range of numbers to be generated, and r is the number of times to repeat the computation of steps 3-6 for each value of n for better measurement of average-time); use the following values of n=10, 50, 100, 500, 1000, 5000, 10000, 50000, 100000 etc. – push the limit of your machine to maximum n that your machine can handle for your implementation.

10) Write a SLC report including at most 2 page write-up on your observations of comparison from step 9; include any graphs/charts that may help express your observation precisely and concisely). This report must be produced using MS-Word or equivalent word-processing applications and must be of professional quality. A link to sample report using SLC has been included in the TopicsCovered page of our class' web presence.

Design Requirements

Code Documentation

For this assignment, you must include documentation for your code as generated by JavaDoc. You should have JavaDoc comments for every class, constructor, and method. By default, JavaDoc should output html documentation to a subfolder within your project (/dist/javadoc). Make sure this folder is included when you zip your files for submission. You do not need to submit a hard copy of this documentation.

Hint: http://stackoverflow.com/questions/4468669/how-to-generate-javadoc-html-in-eclipse

Coding Conventions and Programming Standards

You must adhere to all conventions in the CS 3310 Java coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming files, classes, variables, method parameters and methods. Read the material linked from our class web-pages (in case you can't recall programming styles and conventions from your CS1 and CS2 courses).

Testing

Make sure you test your application with several different values capturing different cases, to make sure it works.

Assignment Submission

- Generate a .zip file that contains all your files, including:
  - Source code files
  - Including any input or output files
- Javadocs
- A single-pdf and source files of your SLC report (step 10)
- Don't forget to follow the naming convention specified for submitting assignments