CMPSC 112
Introduction to Computer Science II
Fall 2016

Laboratory Assignment Four: Studying the Efficiency of Sorting Algorithms

Introduction

The Java programming language supports the declaration and use of arrays, thus allowing for a single variable to hold multiple elements of the same type. Arrays have a wide variety of applications in the areas of video games and scientific simulations. There are also some fundamental operations, such as sorting, that are commonly performed on an array. In this laboratory assignment, you will investigate the sorting of arrays in the Java programming language. In addition to learning more about Java's support for other features, like nested classes, you will keep practicing the use of "timers" to support the experimental evaluation of algorithms. In particular, this laboratory assignment will focus on the performance of sorting algorithms, like bubble sort and insertion sort, as they reorder elements inside of an array. Finally, you will continue to practice using software engineering tools, such as Ant build systems, that help you to compile and run large systems.

Reading Assignment

To start your review of the key features of the Java programming language, please study all of the material in Chapter 1 of the textbook. To further prepare for this assignment, you should also review all of Chapter 2, paying close attention to Sections 2.5 and 2.6 and the discussion of how the Java programming language supports generic methods and nested classes. Finally, you should read Section 3.1, focusing on the content in Section 3.1.2 about sorting and the material in Section 3.1.3 about random number generation. Please also examine the slides that we have discussed during our recent class sessions. If you have questions about this reading assignment or the material that was presented in class, then please see the course instructor or a teaching assistant. If done appropriately, you may post your question to the `#laboratory` channel of our Slack team.

Configuring Git and Bitbucket with your Partner

During this laboratory assignment, you will securely communicate with the Bitbucket.org servers that will host all of our projects. If you are still having trouble using Bitbucket, you should refer to the following web site for additional information: `https://confluence.atlassian.com/display/BITBUCKET/Bitbucket+101`. For this assignment, you are responsible for working with a partner to implement all of the required programs, conduct the necessary experiments, and write a report of your results; you may pick your own partner, provided that you do not work with the same person as you did from a previous assignment. Once you have chosen your partner and collaboratively reviewed and discussed the entire laboratory assignment sheet, please create a Bitbucket repository according to the naming convention `cs112F2016-lab04-<user name one>-<user name two>`. Students who are struggling to create their Bitbucket should see the course instructor.

Building, Running, Repairing, and Enhancing a Java Program

After changing into the "share" repository for this course, please run the "`git pull`" command to download all of the source code and the build system that you will need for this assignment. Once you have been able to successfully compile this program, please take time to review the source code of the provided Java files, making sure that you and your partner can draw a diagram to explain how they are related. You should also carefully study the `build.xml` file so that you can learn about all of the features that it provides. Now, you are ready to run the program by typing the command "`ant run`" in your terminal. What output does this produce? What is the meaning of this output? Do you understand the numbers that the program outputs? You and your partner should examine this output, discussing it so as to ensure that you both understand its meaning.

Even though this program is designed to perform an experiment to evaluate the performance of sorting arrays — as implemented in the bubble sort, insertion sort, and `java.util.Arrays.sort` methods — you will notice that it does not currently produce any output so that you can verify that the sort is working correctly! To ensure that the program sorts arrays in the right way, you should modify some of the existing code so that it produces debugging output for arrays that are small enough so that you can check the correctness of their sorting. Is the program correctly generating and sorting the arrays? If not, you must fix the provided source code. Please see the course instructor or a teaching assistant if you are stuck on this debugging step.

You will also notice that, as it is currently implemented, the `public static Character[] generateRandomCharacterArray(int size)` generates an array of `Character`s in a fashion that is different from how the other random data generators are implemented. Why is the method implemented in this fashion? After you have understood this data generator you should also observe that the `SortArrayExperiment` now has a `discardFirst` variable declared in `main`. What is the purpose of this variable? How does it change the results of the experiment and the conclusions that you might draw about the performance of the three sorting algorithms?

Next, since you and your partner will see that the `SortArrayExperiment` is currently implemented to only sort arrays that contain `Character`s, you should extend it so that it sorts arrays of at least one additional data type, such as `Integer` or `Float`. Once your team has finished implementing this feature, you will likely realize that the source code of the `main` method in `SortArrayExperiment` is becoming long and difficult to maintain and understand. In the final part of this phase of the assignment, you and your partner should apply the principles of object-oriented design to "refactor" the `SortArrayExperiment` so that it is easier to understand. Why do you and your partner think that your program's new design is an improvement over the old one? Whenever possible, you should appeal to the principles developed in Chapter 2 to justify your decisions.

Experimentally Evaluating the Performance of Array Sorting

In this laboratory assignment, you and your partner will conduct an experiment to evaluate the performance of three sorting algorithms. The insertion sort algorithm is described in Section 3.1.2 of your textbook and the bubble sort algorithm is part of the "folklore" of computer science. The third algorithm, provided by the `java.util.Arrays.sort` method, is part of the Java programming language and documented in the online JavaDoc documentation. Before you start conducting an experiment to evaluate the performance of these algorithms, you and your partner should study the

source code and documentation for these three algorithms and formulate a hypothesis as to which you think is likely to be the fastest. As you focus on the iteration constructs and conditional logic used by each of these algorithms, you should develop an intuition about how their structure will influence their performance as the size of the input array varies from very small to very large.

It is also important to remember that the current implementation of the `SortArrayExperiment` has a feature that allows you to discard the first timing from a run of your of your benchmarking framework. You and your partner should carefully discuss this feature and decide if you will use it when conducting your experiments. Additionally, you should also note that the `SortArrayExperiment` uses a variable to control the number of trials for which each of the three algorithms will be run. After conducting some exploratory studies with the `SortArrayExperiment`, your partnership should decide how many trials you will run during your experimental study. Finally, you should conduct experiments to answer the following three research questions.

RQ1: How does the data type of an array influence the performance of the array sorting method? To answer this research question, you should run the `SortArrayExperiment` class for a suitable number of trials and record the timings when the array sort method accepts arrays of type `Character` and at least one of another data type like `Float` or `Integer`; you and your partner are also encouraged to investigate other data types as you deem appropriate. Your response to this research question should leverage the timings and their arithmetic means and standard deviations, particularly focusing on which array data type leads to the fastest and least-variable method for array sorting. You should note the key trends and, additionally, explain why they are evident.

RQ2: Is it faster to reverse an array using a "hand-coded" method or the one furnished by the Java language? To answer this research question, you should run the `SortArrayExperiment` class for a suitable number of trials and focus primarily on whether it is faster to sort an array using the sorting methods provided by this system or the standard method available by calling `java.util.Arrays.sort`. Your response to this research question should leverage the timings and their arithmetic means and standard deviations, particularly focusing on which algorithmic technique leads to the fastest and least-variable method for array sort. Whenever possible, you should note the key trends and, additionally, explain why they are evident in your data set.

RQ3: How does the performance of sorting change as the size of the array increases? To answer this research question, you should systematically run the `SortArrayExperiment` class with progressively larger arrays, looking to see how the execution time of the sorting algorithms increases as the array size increases. In particular, you must run the sorting algorithms for arrays of size 10, 100, 1000, 10000, at at least one size larger than 100000. As you conduct these experiments, please bear in mind that large array sizes may necessitate a long running time for (at least some of) the algorithms. Finally, whenever possible, you should try to formulate the precise relationship between the size of the input array and the time needed by each of the three algorithms to sort the array.

Carefully Review the Honor Code

The Academic Honor Program that governs the entire academic program at Allegheny College is described in the Allegheny Academic Bulletin. The Honor Program applies to all work that is submitted for academic credit or to meet non-credit requirements for graduation at Allegheny. This includes all work assigned for this class (e.g., examinations, laboratory assignments, and the final project). All students who have enrolled in the College will work under the Honor Program.

It is understood that an important part of the learning process in any course, and particularly one in computer science, derives from thoughtful discussions with teachers and fellow students. Such dialogue is encouraged. However, it is necessary to distinguish carefully between the student who discusses the principles underlying a problem with others and the student who produces assignments that are identical to, or merely variations on, someone else's work. While it is acceptable for partners in this class to discuss their programs, data sets, and reports with their classmates, deliverables that are nearly identical to the work of others will be taken as evidence of violating the Honor Code.

Summary of the Required Deliverables

This assignment invites you to submit both a printed and an electronic version of these deliverables:

1. Using the Javadoc standard, a commented version of the three enhanced Java classes.
2. The output from running the final version of the `SortArrayExperiment` program.
3. A short report that responds to these prompts about the provided Java classes:
   (a) What does the use of "`ant run`" command cause the Ant build system to do?
   (b) What is the relationship between the three Java classes used to run the experiments?
   (c) How does `generateRandomCharacterArray` create random arrays of `Character`s?
   (d) How did you enhance the `SortArrayExperiment` to make it easier to understand?
   (e) How do the three Java classes support the effective display of debugging information?
4. A short experiment report that comments on the timing experiments that you conducted. Your experiment report should include, at minimum, the following deliverables:
   (a) An overview of how you collected timing information with the `SortArrayExperiment`.
   (b) A description of the steps that you took to conduct the experiments and collect timings.
   (c) Tables of data that support your responses to the aforementioned research questions.
   (d) A commentary on the observed empirical trends and the reasons why they are evident.
   (e) A statement of the threats to the validity of the conclusions drawn from experimentation.
   (f) A list of the challenges you faced when running the experiments or analyzing the results.
5. A short commentary on the challenges you faced when enhancing the provided source code.
6. A one-page document that explains the tasks completed by each member of your team.
7. An individually submitted document that reports on the work experience with your partner.

Before you turn in this assignment, you must ensure that the course instructor has read access to your Bitbucket repository that is named according to the convention `cs112F2016-lab04-<user name one>-<user name two>`. Please note that each team in the class is responsible for submitting one version of this assignment. Students who have questions about any aspect of this laboratory assignment, including how they should complete it under the structure of the Honor Code, are encourage to schedule a meeting during the course instructor's office hours. Students are also invited to post questions or comments about this laboratory assignment to the `#laboratory` channel in our Slack team; either a teaching assistant or the instructor will answer these questions.