

# Sort Detective Lab

*Credit: Assignment from Sort Detective developed by David B. Levine*

## 1 INTRODUCTION

---

The primary objective of this lab is for you to apply your theoretical knowledge of sorting algorithms to solve a problem of poor user interface design. More specifically, you will be given a program which is designed to measure comparisons, data movements, and execution time for the seven sorting algorithms discussed in class. Unfortunately, the designer of the program did not label the buttons properly. You must apply your understanding of the general properties of the algorithms (and in some cases of the code used to implement them) to determine the proper labeling of the buttons.

The secondary objective of this lab is for you to gain experience writing a concise, but complete analysis of a system.

## 2 BACKGROUND

---

As you know from class, if you double the size of the data set that you give to a quadratic algorithm, it will do four times the work; by contrast, an  $O(n \log n)$  algorithm will do a bit more than twice as much; and, a linear algorithm will do only twice as much work. As you also know, the characteristics of the input data set can affect the expected performance of many of our sorting algorithms. Before you begin the lab, you should review the expected performance of the algorithms on various data sets.

The sorting algorithms under study include (in no particular order): bubble sort, insertion sort, merge sort, quick sort, heap sort, and selection sort.

## 3 INSTRUCTIONS - WARNING: READ ALL OF THE INSTRUCTIONS BEFORE BEGINNING!

---

1. Begin by copying the Sort Detective application from blackboard. Execute it and play with it a bit. Notice that the button names do not give any indication which sort they will execute. Notice also, that if you create a small list, then that list is shown to you in the window.
2. Devise a plan which will enable you to match the particular algorithms to the button names. Hint: It may make sense to try to divide the sorts into initial groups and then to work on each group separately. Divide and conquer: it works for algorithms and it can work here, too!
3. Execute your plan, taking careful notes as you go.
4. Describe the results of your experiment in a summary document. Begin with a summary of the matching and then show the rationalization process that justifies it.

## 4 A NOTE ON WRITING

---

There is no coding in this lab. Thus, you should expect that a **significant** portion of the lab grade for this lab will be determined by the quality of the writing of the report. This includes the completeness of the report, the clarity (and grammar) of the writing, and general presentation. It is possible to receive a poor grade even if you match all sorts correctly due to sloppy writing.

Some of the sorts are very difficult to distinguish. A carefully outlined experiment may compensate for an error in these cases if the writing makes it clear that your conclusions/guesses are substantiated by the data.

Finally, remember that your report needn't detail every experiment you ran. Rather, it should give sufficient information to justify your conclusions. It is possible to write a very short report that is completely correct if your experiments are well-chosen. After you learn the matching, you might consider whether there was a shorter way to arrive at your conclusion!

## 5 DELIVERABLES

---

The Final report from Step 4.

# Example

You will need much more analysis for the assignment but the following gives an idea of what kind of analysis will be required.

## 5.1.1 General Results

Button Name	Searching Algorithm
Algorithm 1	Binary Search
Algorithm 2	Sequential Search

## 5.1.2 Rationale

We chose to search data sets of size 500, 1000, and 2000, looking for the last item each time. Algorithm 1 took 500, 1000, and 2000 comparisons each time, doubling the number of comparisons each time the size of the searchable list doubled. Algorithm 2 took many fewer comparisons and this number increased by only one each time we doubled the size of the searchable list.

Since sequential search is linear, we conclude that Algorithm 2 is SEQUENTIAL SEARCH.

Then, by process of elimination, we conclude that Algorithms 1 is BINARY SEARCH.