Computer Science 2XC3: Lab 2 and 3

Dr. Vincent Maccio

Please read the following carefully. There are no completion/participation grades in labs. However, the TAs are there to guide and aid you in your tasks.

# Purpose

This lab focuses on implementing, analysing, and optimizing some traditional sorting algorithms. This lab will span two weeks. In the first week you will cover the following (but not limited to):

* Compare the runtimes of Bubble, Selection, and Insertion sorts
* Implement variations of each Bubble and Selection sort and run experiments to determine how much (if any) improvements you observe
* Analyse the performance of these algorithms under specific cases such as short lists and if the list are “near sorted”

During the second week you will cover the following (but not limited to):

* Compare the runtimes of Merge, Quick, and Heap sorts
* Implement variations on Merge and Quick sort and run experiments to determine how much (if any) improvements you observe
* Analyse the performance of these algorithms under specific cases such as short lists and if the list are “near sorted”
* Determine if a “hybrid” search strategy would be beneficial
* Implement your sorting algorithms via a strategy pattern (we’ll see how the workload is)

# Part 1

Throughout this lab you will be creating and ultimately submitting a lab report. Your lab report should look professional and complete. It should include the following:

* Title page
* Table of Content
* Table of Figures
* An executive summary highlighting some of the main takeaways of your experiments (this can be presented in bullet form if you wish)
* A clearly marked section for each experiment/exercise outlined in this lab
* An appendix explaining to the TA how to navigate your code (for example, which .py file to find which implementation/experiment in)

For each experiment, include a clear section in your lab report which pertains to that experiment.

## Experiment 1

In the file bad\_sorts.py posted alongside this document, you will find implementations of Bubble, Insertion, and Selection sort. Run suitable experiments to compare the runtimes of these three algorithms. In bad\_sorts there is a create\_random\_list function which may be useful. In your report this section should include:

* An explicit outline of the experiments you ran. That is, list length values, how many “runs”, etc.
* A graph of *list length vs time* displaying the three curves corresponding to the three “bad” sorting algorithms
* A brief discussion and conclusion regarding the results. A few sentences are fine here.

## Experiment 2

Implement the following two algorithm variations.

* In lecture we saw that we could improve insertion sort by keeping track of the value we are inserting, finding the location that we want to insert it into, and then appropriately shifting the remaining values, rather than “swapping” it down to where it should go. Review the lecture material if you are unfamiliar with this or ask the TA to explain it. Apply the same general approach to Bubble sort. This will be slightly more complicated since in bubble sort you will need to potentially insert many values and shift things appropriately during a single iteration. Name this implementation of Bubble Sort bubblesort2().
* Instead of having selection sort keep track of the minimum value during a single iteration and positioning it accordingly, have it keep track of the min and max value of a single iteration and position both values accordingly. Note, your loop *boundaries* should be updated accordingly as well. Ask the TA for clarification if things are not clear.

Run experiments where you compare the original Selection and Bubble Sort runtimes to their potential improvements. In your report this section should include:

* An explicit outline of the experiments you ran. That is, list length values, how many “runs”, etc.
* Two separate *list length vs time* graphs:
  + One comparing the original Bubble Sort and its variation
  + One comparing the original Selection Sort and its variation
* A brief discussion and conclusion regarding the results. A few sentences are fine here.

## Experiment 3

In bad\_sorts.py you will find a create\_near\_sorted\_list(length, max\_value, swaps) function. This creates a random list of length *length* of values between 0 and *max\_value*. Furthermore, it will make a number of random swaps equal to *swaps.* For example, if swaps = 0, the list will be perfectly sorted. If

swaps = length\*log(length) / 2

the list will be statistically indistinguishable from a randomly generated list. Do not worry about understanding the above result – just trust it for now.

Run an experiment where you compare the runtimes of the three bad sorting algorithms (or their improvements, your choice) vs how many random swaps are made on sorted list. For this experiment fix the list length to be constant at 5000 (or another reasonable value – nothing too small). In your report this section should include:

* An explicit outline of the experiments you ran. That is, the fixed list length, number of runs, swaps, etc.
* A graph of *swaps vs time* displaying the three curves corresponding to the three “bad” sorting algorithms. Choose the range of swaps wisely here. If interesting things do not occur for values past a certain threshold, do not feel obligated to include those on the graph.
* A brief discussion and conclusion regarding the results. A few sentences are fine here.

# Part 2

Coming soon…

# Grading and Submission

Your group will submit the following documents to Avenue:

* report.docx (or .pdf, or whatever – as long as a reasonable person can open it)
* code.zip (all your source code pertaining to the lab – including experiment code)

In addition to the grade allocations below, your report may lose up to 20% of the final grade for not looking professional, having formatting/style issues, graphs presented in a messy manner, etc. Moreover, you may lose grades for not including elements explicitly mentioned in the Part 1 section of this document. Find a rough grade breakdown below:

Part 1

|  |  |
| --- | --- |
| Experiment 1 | 10% |
| Experiment 2 and corresponding implementations | 20% |
| Experiment 3 | 10% |

Part 2

|  |  |
| --- | --- |
| Experiment 4 | 10% |
| Experiment 5 and corresponding implementations | 20% |
| Experiments 6 and 7 | 20% |
| Strategy Pattern | 10% |