

COSC 282 Lab 3.1 Sort Algorithm Tester GUI

In this lab we'll be modifying the Testing GUI to allow us to test sorting algorithms. We'll be working in teams of 4 students, one from each of the teams from lab 2.3. This will allow each team to start with four complete working code bases to evaluate, learn from, and modify.

Preparation and Planning

Begin by carefully reading through the lab requirements. Then spend some time as a group examining the four GUIs from lab 2.3. With one team member keeping notes, discuss the following:

1. What design elements or choices do all four GUI prototypes have in common?
2. What aspects serve to enhance visual appeal?
3. What aspects contribute to ease of operation?
4. Review and discuss the changes described below. Based on the four prototypes, sketch a GUI for a sort algorithm tester. You might, for example, choose one of the four as your starting point; or you might combine elements of several of them.

Modifications to GUI and Code

As a team, and with one team member keeping notes, discuss and design the modifications needed to allow for the following:

1. When run, the program should perform a series of tests using arrays of increasing size. The arrays should initially be randomly populated; they should then be sorted, keeping track of the average number of comparisons and average number of copies at each array size.
2. Bubble sort, selection sort, and insertion sort should all be implemented within the same class.
3. As before, the GUI should allow the user to set various parameters for the testing.
4. Add a controller (or controllers) to allow the user to select which sort algorithm to use.
5. Update the output to include all relevant parameters in the output file, along with the series of testing data.

Finally, once you've selected your final layout, implement it in Java and test it thoroughly. Each team will present and demonstrate their final product to the entire class.

Your submission will consist of the following:

1. A formal lab report (PDF format, one per group):
 - a. Cover page with lab number and title, course number, date, names of team members, and their email addresses.
 - b. An overview that summarizes the work completed.
 - c. A breakdown of work contributed by each team member.
 - ~~d. Summary of the discussion on which controls to use.~~
 - ~~e. The candidate design proposals by each team member.~~
 - f. Summary of the discussion on GUI designs.
 - g. A listing of file names / Java classes included in the final code base.
 - h. Screenshots of successful runs of the program.

2. Your code (one per group), which should be
 - a. Fully commented
 - i. Include a header comment in every file.
 - ii. Include a class comment for every class.
 - iii. Include @author and @version tags in every class.
 - iv. Include @param and @return as appropriate in all methods.

 - b. Formatted and indented in a consistent manner. If using Eclipse, I recommend using the setting Preferences::Java::Code Style::Formatter:: Eclipse [built-in]. Once that it set, you can keep your code neat by using the command Source::Format regularly.

3. A summary of what you learned doing this lab (one per person).

Submission instructions:

1. Team Leader (first student alphabetically)
 - a. in Sakai, submit zipped archive with lab report & code for the team.
 - b. In the text submission box, indicate the names of the team members.
 - c. verify that submission is correct, then add "I confirm that this is my full and final submission" to the text submission box
 - d. email a copy of the submission to the team members (cc me) stating that the lab has been submitted

2. Other team members
 - a. in Sakai, submit your individual learning summary paragraph.
 - b. in the text submission box, indicate the names of the team members.