Stress Tests; Heapsort

12.1 Goals: StressTests - Timing Tests

Your job for this lab is to be an algorithm detective. The program we provide allows you to run one of six different sorting algorithms on data sets of different sizes using three different `Comparator`s to define the ordering of the data. When you run the program, the time that each of these algorithms took to complete the task is shown in the console.

To run the program you need to do the following:

- Create a new *Java Project* in Eclipse (e.g. *SortingLab*).
- Add `sorting.jar` to the build-path.
- Create a new *package* in Eclipse and give it the name `student`. Note the new directory that is created. Packages help separate files classes, and allow others to import your library.

  The Zip file for this Lab contains two files (`SortingHeapSort.java` and `Heapsort.java`) in the `student` that you should add to the `student` package (put them in the right folder).
- Place the `citydb.txt` file in your project directory (the one with the `src` and `bin` directories for *SortingLab*).
- Add a new *Run Configuration* named it *SortingTests* then click on the button to *Select main*. As the main class select `sorting.Interactions`. When you run it, you should see a GUI with several check-boxes and radio buttons.
- To run the timing tests the program goes through three steps:

  1. It reads in the data for the 29470 cities from the `citydb.txt` file. The button *FileInput* opens a file chooser dialog, so you can select your `citydb.txt`.

2. You must click the *TimerInput* button to select which algorithms, `Comparator`s, and size data to be used in the tests. Start with the smaller tests and see how the program behaves before you decide to run the larger tests.

   The last algorithm choice is *heapsort* (which you will implement). The two files in the `student` package provide only hooks for the program — the `heapsort` method in the `Heapsort` class just returns the original unsorted `ArrayList`.
3. Now you can run the actual tests by hitting the `RunTests` button.

   You may repeat the last two steps as many times as you want to, though you will need to restart the program if/when you modify your source files.

Exploration

Now try to answer some of the following questions. **Finish this work as a part of the Assignment 12.**

Run the program a few times with small data sizes, to become familiar with its behavior and the approximate times. Then run experiments and try to answer the following questions:

1. Which algorithms run in mostly quadratic time, i.e. $O(n^2)$?
2. Which algorithms run in mostly $O(n.log_n)$ time?
3. Which algorithms use functional style (immutable) `Cons` lists? Explain how you came to that conclusion.
4. Which algorithm is *selection sort*? Justify your answer.
5. Why is there a difference when the algorithms use a different `Comparator`? Where do you see this difference most prominently and why?

Copy the timing results into a spreadsheet. You may save the result portion in a text editor with a `".csv"` suffix and open it in *Excel* (or some other spreadsheet of your choice). You can now study the data and represent the results for different sizes as a chart. Do so for at least three algorithms,

where there is one of each — a quadratic algorithm and a *linear-logarithmic* algorithm.

Produce a report with a paragraph that explains what you learned, using the *Excel* charts to illustrate this.

Your report should have a professional look – typed, computer generated charts, reasonably organized. It does not have to be more than 2 pages long, one page is OK.