A1-Structured Insertion Sort

Stuart Thiel

July 3, 2022

Introduction

This assignment is worth **5%** of your grade. This is an **individual assignment** (so individual, I put *your* student ID on it) and you should not share your assignment with anyone else. The **assignment is due July 10th at Midnight, Montreal time**.

All submissions must go through EAS: `https://fis.encs.concordia.ca/eas/`

After you submit, go back to check EAS to confirm that you just uploaded your assignment!

Q1) Structured Insertion Sort

Your program should be called `StrInSort.java` and it should define a single class, StrInSort. It's main method will consider one argument, the name of a file containing your input data. Input data should be positive integers separated by spaces.

Your goal is to take a "structuring pass" over the data before performing a regular insertion sort. Your structuring pass will look for runs, starting from the beginning of the array and moving left to right. Whenever you find a run in this manner that is in *ASCENDING* order you should reverse it. Keep track of how many times you reverse values in this fashion. Also keep track of any runs in *DESCENDING* order of length 3.

When you go to sort your data, make sure you sort it in *DESCENDING* order.

Take a look at the example output on the next page to see how to present your results and note what things you have to look out for while writing your code. DebugRunner can be pretty picky, so you want to make sure you're pretty close. Don't do wasteful swaps or compares. Use the same names in your own output. Note the use of singular tense when outputting results about a single occurrence of something. These are just small programming details, but I want to remind you to keep an eye out for details.

1.1 Evaluation

You will notice that there were some other files in the zip containing this assignment PDF. There should also be a file called `DebugRunner.class` and one caled `config.xml`. This version also includes a number of test inputs. `StrInSort.class` file, compiled from the Java source you wrote, in the same folder as them, you can run `java DebugRunner` to get some feedback. You'll want to run this from the command-line. Ask your tutors for help in doing this if you can't figure it out. Most assignments will include this and the config.xml file will partially mark your assignment so you can figure out how far along you are, though the marker will have a more comprehensive `config.xml`, so make sure you follow assignment instructions carefully!

Structured Insertion Sort Example

You might type:
java StrInSort inputs1.txt

`inputs1.txt` might look like:

```
22 80 89 60 52 19 61 77 15 62 30 80 79 20 81 32 82 13 77 91 54 42 47 91 18
```

The output would look something like:

```
22 80 89 60 52 19 61 77 15 62 30 80 79 20 81 32 82 13 77 91 54 42 47 91 18
We sorted in DEC order
We counted 1 DEC run of length 3
We performed 6 reversals of runs in ASC order
We performed 24 compares during structuring
We performed 192 compares overall
We performed 151 swaps overall
91 91 89 82 81 80 80 79 77 77 62 61 60 54 52 47 42 32 30 22 20 19 18 15 13
```

Q2) Theory: Qualify This Structuring

How did the structuring pass you performed, specifically the reversals chosen, affect swaps and comparisons? Was anything else affected? Answer in less than 100 words.

Q3) Theory: Size of Runs

How do you feel the size of the specific runs you recorded (*DESCENDING* order of length 3) impacted performance? Answer in less than 100 words.

Q4) Theory: Doubly Linked Lists

What would implementing this as a Doubly Linked List do? How would the specified structuring affect results? Answer in less than 100 words.