Joanna Klukowska
joannakl@cs.nyu.edu

Programming Project 6:
NYPD Motor Vehicle Collisions Analysis - Revisited

Due date: Dec. 11, 11:55 PM EST.

You may discuss any of the assignments with your classmates and tutors (or anyone else) but all work for all assignments must be entirely your own. Any sharing or copying of assignments will be considered cheating. If you get significant help from anyone, you should acknowledge it in your submission (and your grade will be proportional to the part that you completed on your own). You are responsible for every line in your program: you need to know what it does and why. You should not use any data structures or features of Java that have not been covered in class (or the prerequisite class). If you have doubts whether or not you are allowed to use certain structures, just ask your instructor.

In this project you will revisit your project 1 implementation of a program that used the NYPD motor vehicle collision data. A version of the solution to the original project is provided (it has been slightly modified from the specification of the original project). You should use that implementation as the basis for your project 6 solution.

Please, see project 1 for the details of tasks and implementation requirements.

Objectives

The goal of this programming project is for you to master (or at least get practice on) the following tasks:

- using/modifying existing code
- selecting data structures appropriate for given tasks
- writing Java programs

The Program Input and Output

You should not change this part of the implementation.

There are three data sets posted on the course website that you should be using for this project. The largest of them contains all collision data for the entire NYC from July 1, 2012 till June 30, 2015 (three years total). The medium size data set contains all collision data for the entire NYC from July 1, 2013 till June 30, 2015 (two years total). Finally, the smallest data set contains all collision data for the entire NYC from July 1, 2014 till June 30, 2015 (one year total).

For testing purposes, you may wish to use smaller data sets.

Computational Task

The posted program stores all the `Collision` objects in an `ArrayList` of `ZipCodeList` objects. The `ZipCodeList` object, in turn, stores all `Collision` objects from a particular zip code in an `ArrayList` of collisions.

At this point in the semester, you should realize that this is not an optimal way of organizing the data. Your task for this project is to modify how the Collision objects are stored in memory when the program is running in order to improve the time performance of the program.

Joanna Klukowska
joannakl@cs.nyu.edu

You do not need to provide implementation of any data structures yourself. You should use the implementations provided by Java API. You should only use the data structures that we discussed in class during this semester. Here is the list with links to Java API interfaces and classes:

- linked list https://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html
- stack https://docs.oracle.com/javase/8/docs/api/java/util/Stack.html
- queue https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html (any of the implementing classes)
- (self-balancing) binary search tree http://docs.oracle.com/javase/7/docs/api/java/util/TreeSet.html
- priority queue https://docs.oracle.com/javase/8/docs/api/java/util/PriorityQueue.html
- hashtable https://docs.oracle.com/javase/8/docs/api/java/util/Map.html (any of the implementing classes)

You are not supposed to use all of the above data structures. Your task is to decide which of them could improve the time performance of the program and use those.

You may wish to try several different changes. In your final program you should leave only the changes that resulted in a better time performance of the program. All other changes should be described in the written report (see below).

Written Report: Performance Analysis

You should write a 1-2 page report that summarizes the results of this project. Your report should include justification for why particular data structures are appropriate and others are not. You may wish to include time comparison results for different sizes of data sets that you obtain with the provided version of the code and with your modified version of the code. If you tried some of the data structures and they did not turn out to be appropriate, mention that in your report.

If you decide to include graphs and tables, those do not count towards the 2-page limit.

Program Design

You should only change parts of the design of this program that are related to change in data structures. Overall program design should remain the same.

Programming Rules

You should follow the rules outlined in the document *Code conventions* posted on the course website at http://cs.nyu.edu/~joannakl/cs102_f15/notes/CodeConventions.pdf.

You must document all your code using Javadoc. Your class documentation needs to provide a description of what it is used for and the name of its author. Your methods need to have description, specification of parameters, return values, exceptions thrown and any assumptions that they are making.

A class's data fields and methods should not be declared `static` unless they are to be shared by all instances of the class or provide access to such data.

You must use the provided implementation of the program and modify only the parts that change the storage of Collision objects.

Grading

Make sure that you are following all the rules in the **Programming Rules** section above.

If your program does not compile or crashes (almost) every time it is ran, you will get a zero on the assignment.

Joanna Klukowska
joannakl@cs.nyu.edu

If the program does not adhere to the specification, the grade will be low and will depend on how easy it is to figure out what the program is doing.

30 points choice and use of data structures,

40 points written report with analysis of the observations,

10 points use of existing program structure,

20 points proper documentation and program style (for the parts of the code that you modify; you should add your name as the author for the classes that you modify).

How and What to Submit

Your should submit all your source code files (the ones with .java extensions only) and your report (in PDF format) in a single **zip** file to NYU Classes. If you are unsure how to create a zip file or how to get to your source code files, you should ask long before the project is due.

If you wish to use your (one and only) freebie for this project (one week extension, no questions asked), then complete the form at `http://goo.gl/forms/YFDVBlscEB` **before the due date for the assignment**. All freebies are due seven days after the original due date and should be submitted to NYU Classes.