Programming assignment 3

Getting started

All your submissions should be implemented in Java. If you do not have a Java development environment, we recommend that you either use the Visual Studio Code Java extensions or Jetbrains IntelliJ. If you use IntelliJ, do not forget to register as a student, so you get access to the full versions. If you run Linux, macOS, or WSL, and want more control over your JDK, we recommend SDKMAN!

Note that you must solve all of the programming problems in Java. Kotlin or Scala are also ok, but you might not get as much support. You are allowed to use Python (or any other suitable tool) to analyse the results, plot graphs, and so on.

Problems

Problem 1

Implement a hash table that uses quadratic probing to handle conflicts. Use strings to test it. Using your own (poor) hash function is probably easier to ensure that conflict handling works as expected.

Problem 2

Use the hash table from Problem 1 to store information about vehicles. Create a class that contains at least the license plate number and a hash function that uses that number to determine which bucket the object should be placed in.

Conduct an experiment to analyze how well your hash function works. You can, for example, study the number of conflicts and the offset.

Problem 3

Implement Insertsort.

Problem 4

Implement heapsort.

Problem 5

Implement Quicksoft. Use median of three to determine the pivot value. Your implementation should take a parameter, `depth`, which determines at which recursion depth Quicksort should swap to Heap- or Insertsort.

Conduct an experiment to determine recommended for `depth` and whether Heap- or Insertsort should be used.

Problem 6 (optional, for higher grade)

Implement an iterative version of Mergesort, i.e., a version that does not use recursion. Compare it to a recursive implementation. Which is faster on average.

Problem 7 (optional, for higher grade)

Implement Shellsort and experiement with different Gap sequences. Implement at least 2-3 differen seqences. Use Wikipedia's article as a starting point for various gap sequences.

Submission guidelines

Submit your solutions as a single zip-file via Moodle no later than 17:00 on October 21, 2022 (cut-off 24:00 October 23). This is an individual assignment. Your submission should contain well-structured and organized Java code for the problems with a README.txt (or .md) file that describes how to compile and run the Java programs. You should also include a report that describes your experiment setup and findings from problems 2 and 5. If you solve problem 6 and 7, the report should also contain your experiment setup and findings from that problem. The report should be a PDF file.