Algorithmen und Datenstrukturen

Dr. M. Lüthi, Dr. G. Röger
Frühjahrssemester 2018

Universität Basel
Fachbereich Informatik

Übungsblatt 1

Abgabe: 9. März 2015

Für dieses (und folgende) Übungsblätter benötigen Sie *Java*. Bitte installieren Sie Java auf Ihrem System. Unter Ubuntu geht dies zum Beispiel mit `sudo apt install openjdk-8-jdk`.
Für andere Betriebssysteme finden Sie die notwendigen Informationen unter
`https://www.java.com/en/download/manual.jsp`.

Im ADAM-Workspace der Vorlesung finden Sie die Datei `blatt01.zip`. Bitte laden Sie sie herunter und entpacken Sie sie. Im Verzeichnis `blatt01` befinden sich die Skripte `gradlew` (für UN*X Betriebssysteme wie Linux, Mac OS, ...) und `gradlew.bat` (für Windows), die als Wrapper für die Software *Gradle* (`https://gradle.org/`) dienen. Beim ersten Ausführen lädt das Skript Gradle herunter und speichert es auf Ihrem System (z.B. bei Linux unter `~/.gradle`).

Aufgabe 1.1 (Insertionsort, 2+1+1+2 Punkte)

(a) In `blatt01/src/main/java/InsertionSort.java` finden Sie eine unvollständige Implementierung von Insertionsort für int-Arrays. Datei `blatt01/src/test/java/SortTests.java` enthält erste Unit Tests für die gewünschte Funktionalität. Ein Test prüft die Funktionalität für leere Arrays, der andere für ein nicht-leeres Beispielarray.

Gehen Sie in Verzeichnis `blatt01` und rufen Sie `./gradlew test` bzw. `gradlew.bat test` auf. Lesen Sie die Ausgabe: Test `insertionSortShouldCorrectlySortNonemptyIntArray` ist fehlschlagen.

In `blatt01/src/main/java/Main.java` finden Sie eine Main-Methode, die die Insertionsort-Implementierung für ein Beispiel aufruft und das (aktuell noch nicht) sortierte Array ausgibt. Sie können die Main-Methode mit `./gradlew run` bzw. `gradlew.bat run` (aus Verzeichnis `blatt01`) aufrufen.

Vervollständigen Sie die Implementierung von `InsertionSort.sort(int[] array)`. Danach sollte Gradle mit `test` ohne Fehler durchlaufen (BUILD SUCCESSFUL).

(b) Neben der korrekten Funktionalität, ist auch ein durchgängiger, gut lesbarer Stil des Codes wichtig. In dieser Vorlesung verwenden wir den sogenannten google-Stil, der unter `https://google.github.io/styleguide/javaguide.html` beschrieben ist. Rufen Sie Gradle mit `check` auf, um einen Grossteil der Vorgaben automatisiert zu testen. Daraufhin finden Sie unter `blatt01/build/reports/checkstyle` html-Dateien, mit denen Sie das Ergebnis etwas schöner im Browser anzeigen können. Überarbeiten Sie Ihren Code bis Sie keinen Fehler mehr bekommen.

(c) Für Ihre nächste Anwendung wollen Sie nicht int-Arrays, sondern Integer-Arrays sortieren. Erweitern Sie dazu zunächst die Klasse `SortTests.java` um entsprechende Tests `insertionSortShouldCorrectlySortEmptyIntegerArray` und `insertionSortShouldCorrectlySortNonemptyIntegerArray`, die die Funktionalität einer hypothetischen Methode `InsertionSort.sort(Integer[] array)` testet.

(d) Da Sie InsertionSort nicht immer wieder für neue Typen implementieren wollen, werfen Sie zunächst einen Blick auf das Interface `Comparable`, das auch von der Klasse `Integer` implementiert wird. Das Interface fordert für jede implementierende Klasse `T` die Methode `int compareTo(T o)`, bei der der Rückgabewert von `a.compareTo(b)` angibt, ob `a` kleiner, gleich oder grösser `b` ist. Erweitern Sie die Klasse `InsertionSort` um eine Methode `public`

`static <T extends Comparable<T>> void sort(T[] array)`, die Insertionsort für beliebige Comparable-Arrays implementiert. Sie können auch gerne in der Main-Methode ein entsprechendes Beispiel mit einem Integer-Array einkommentieren. Stellen Sie sicher, dass Gradle mit `test` und `check` keine Probleme mehr findet.

Aufgabe 1.2 (Mergesort, 1 + 2 + 1 Punkte)
Im Rahmen dieser Aufgabe implementieren Sie Top-Down-Mergesort für int-Arrays.

(a) Erweitern Sie zunächst die Testklasse `SortTests.java` um geeignete Methoden `mergeSortShouldCorrectlySortEmptyIntArray` und `mergeSortShouldCorrectlySortNonemptyIntArray`, die die Methode `MergeSort.sort(int[] array)` testen.

(b) Vervollständigen Sie die Methode `MergeSort.merge(int[] array, int[] tmp, int lo, int mid, int hi)`.

(c) Vervollständigen Sie die Methoden `MergeSort.sort(int[] array)` und `MergeSort.sort(int[] array, int[] tmp, int lo, int hi)`.

Stellen Sie sicher, dass Ihr Code alle Tests und den Stylecheck fehlerfrei besteht.

Die Übungsblätter dürfen in Gruppen von zwei Studierenden bearbeitet werden. Bitte benennen Sie das Verzeichnis `blatt01` *in* `blatt01-<name1>-<name2>` *(mit Ihren Namen) um. Führen Sie in dem Verzeichnis Gradle mit* `clean` *aus, zippen Sie das Verzeichnis und laden Sie die gezippte Datei unter* `https://courses.cs.unibas.ch/` *hoch.*