TU Berlin
Software and Business Engineering
Prof. Dr. Ingo Weber
Dr. Tobias Heindel
Richard Hobeck
Martin Grambow

WiSe 2020/21
Aufgabe 7
Block 3
4. Januar 2021

Abgabetermin: 10. Januar 2021

Aufgabe 7 *Einführung in die Programmierung mit JAVA*

Hinweise

- Die Abgabe dieser Übungsaufgaben muss bis spätestens Sonntag, den 10. Januar 2021 um 23:59 Uhr im ISIS-Kurs erfolgt sein. Es gelten die Ihnen bekannten Übungsbedingungen.
- Lösungen zu diesen Aufgaben sind als gezippter Projektordner abzugeben. Eine Anleitung zum Zippen von Projekten finden Sie auf der Seite des ISIS-Kurses. *Bitte benutzen Sie einen Dateinamen der Form* `VornameNachname.zip`.
- Bitte beachten Sie, dass Abgaben im Rahmen der Übungsleistung für die Zulassung zur Klausur relevant sind. Durch Plagiieren verwirken Sie sich die Möglichkeit zur Zulassung zur Klausur in diesem Semester.

Erstellen Sie für diese Hausaufgabe ein neues Projekt mit Namen Aufgabe7.

Aufgabe 7.1 *Zusammenfügen von zwei sortierten Arrays* (1 Punkt)

Erstellen Sie eine Klasse `ArrayCoalescing` mit einer Klassenmethode `merge`, die zwei aufsteigend sortierte `int`-Arrays als Formalparameter hat und ein `int`-Array als Rückgabe; die Rückgabe soll die sortierte Version der Konkatenation der zwei `int`-Arrays sein.

Aufgabe 7.2 *Implementieren von `Integer`-Listen* (1 Punkt)

Erstellen Sie eine Klasse `SinglyLinkedIntegerList` und implementieren Sie die Methoden `append` und `getValue` wie auf den Folien der Vorlesung beschrieben—*allerdings* mit `Integer`-Objekten anstelle von allgemeinen „`Object`en“.

Aufgabe 7.3 *Zusammenfügen von zwei sortierten Listen* (1 Punkt)

Erstellen Sie eine Klasse `ListCoalescing` mit einer Klassenmethode `merge`, die zwei aufsteigend sortierte `Integer`-Listen als Formalparameter hat und eine `Integer`-Liste als Rückgabe; die Rückgabe soll die sortierte Version der Konkatenation der zwei `Integer`-Listen sein.

Tipp Es reicht die Methoden `append` und `getValue` zu benutzen.

Hinweis Es geht natürlich effizienter.

Aufgabe 7.4 *Randomisierte binäre Suche* (2 Punkte)

Erstellen Sie eine Klasse `RandomizedBinarySearch` mit einer Klassen-Methode `search`, die ein `int`-Array und einen `int`-Wert als Formalparameter bekommt und ein `Integer`-Objekt zurück gibt; die `search`-Methode soll eine Variante der binären Suche implementieren, in der die Teilung des Suchbereichs bzgl. eines zufälligen Indexes vorgenommen wird (und dieser Index liegt meistens *nicht* in der Mitte). Die Rückgabe soll entweder den ersten Index angeben, an dem

```
import java.lang.Math;
import java.time.Duration;
import java.util.Arrays;
import java.time.Instant;

public class RandomizedBinarySearch {

    public static Integer search(int[] a, int i){...}

    public static void main(String[] args) {

        int[] a = new int[10000000];

        // wait for user to interrupt
        while(true) {
            for (int i = 0; i < a.length; i++) {
                a[i] = (int) Math.round(Math.random() * 1000000000);
            }

            int i = a[0];

            Arrays.sort(a);

            Instant start;
            Instant finish;
            start = Instant.now();
            search(a,i);
            finish = Instant.now();

            Instant begin;
            Instant end;

            begin = Instant.now();
            Arrays.binarySearch(a,i);
            end = Instant.now();

            System.out.println("randomized takes " + Duration.between(start,finish));
            System.out.println("reference takes  " + Duration.between(begin,end));
        }
    }
}
```

Abbildung 1: Test für die randomisierte binäre Suche

sich der Wert befindet im Array, oder die `null`-Referenz (wenn der zweite Formalparameter nicht im Array vorkommt).
Benutzen Sie die `main`-Methode aus Abbildung 1, um mögliche Laufzeiten für zufälligen Arrays experimentell zu bestimmen.

Aufgabe 7.5 *»Insertionsort« mit Listen* (2 Punkte)

Erstellen Sie eine Klasse `SortedIntList` für Listen, die (aufsteigend) sortiert sind.

- Schreiben Sie eine Methode `insertInOrder`, die lediglich einen `int`-Wert als Formalparameter bekommt; diese fügt diesen `int`-Wert an der ersten möglichen Position in die Liste ein.
- Schreiben Sie einen Konstruktor `SortedIntList`, der ein `SinglyLinkedIntegerList`-Objekt als Formalparameter bekommt und dann alle `int`-Werte der Elemente der Liste der Reihe nach einfügt, und zwar mittels der `insertInOrder`-Methode.

Hinweis Ja, Sie haben dann Insertionsort für `Integer`-Listen implementiert![1]

[1] Siehe auch die Seite auf Wikipedia.