

Einzelprüfung „Theoretische Informatik / Algorithmen / Datenstrukturen (nicht vertieft)“

Einzelprüfungsnummer 46115 / 2017 / Frühjahr

Thema 2 / Aufgabe 4

(händisch sortieren, implementieren, Komplexität)

Stichwörter: Bubblesort

Bei Bubblesort wird eine unsortierte Folge von Elementen a_1, a_2, \dots, a_n , von links nach rechts durchlaufen, wobei zwei benachbarte Elemente a_i und a_{i+1} getauscht werden, falls sie nicht in der richtigen Reihenfolge stehen. Dies wird so lange wiederholt, bis die Folge sortiert ist.

- (a) Sortieren Sie die folgende Zahlenfolge mit Bubblesort. Geben Sie die neue Zahlenfolge nach jedem (Tausch-)Schritt an: 3, 2, 4, 1

Lösungsvorschlag

```
3  2  4  1  Eingabe
3  2  4  1  Durchlauf Nr. 1
>3  2< 4  1  vertausche (i 0<>1)
2  3 >4  1< vertausche (i 2<>3)
2  3  1  4  Durchlauf Nr. 2
2 >3  1< 4  vertausche (i 1<>2)
2  1  3  4  Durchlauf Nr. 3
>2  1< 3  4  vertausche (i 0<>1)
1  2  3  4  Durchlauf Nr. 4
1  2  3  4  Ausgabe
```

- (b) Geben Sie den Bubblesort-Algorithmus für ein Array von natürlichen Zahlen in einer Programmiersprache Ihrer Wahl an. Die Funktion `swap (index1, index2)` kann verwendet werden, um zwei Elemente des Arrays zu vertauschen.

Lösungsvorschlag

```
public class BubbleSort {

    public static void swap(int[] array, int index1, int index2) {
        int tmp = array[index1];
        array[index1] = array[index2];
        array[index2] = tmp;
    }

    public static void bubblesort(int[] array) {
        boolean swapped;
        do {
            swapped = false;
            for (int i = 0; i < array.length - 1; i++) {
                if (array[i] > array[i + 1]) {
                    swap(array, i, i + 1);
                    swapped = true;
                }
            }
        } while (swapped);
    }
}
```

}

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/examen/examen_46115/jahr_2017/fruehjahr/BubbleSort.java](https://github.com/bschlangaul/examen/examen_46115/jahr_2017/fruehjahr/BubbleSort.java)**Test**

```
import static org.junit.Assert.assertEquals;

import org.junit.Test;

public class BubbleSortTest {

    @Test
    public void teste() {
        int[] array = new int[] { 3, 2, 4, 1 };
        BubbleSort.bubblesort(array);
        assertEquals(1, array[0]);
        assertEquals(2, array[1]);
        assertEquals(3, array[2]);
        assertEquals(4, array[3]);
    }
}
```

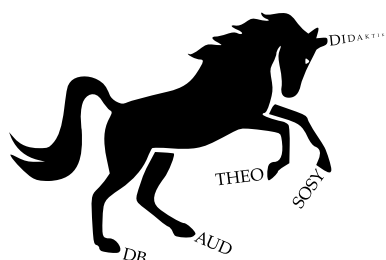
Code-Beispiel auf Github ansehen: [src/test/java/org/bschlangaul/examen/examen_46115/jahr_2017/fruehjahr/BubbleSortTest.java](https://github.com/bschlangaul/examen/examen_46115/jahr_2017/fruehjahr/BubbleSortTest.java)

- (c) Geben Sie eine obere Schranke für die Laufzeit an. Beschreiben Sie mögliche Eingabedaten, mit denen diese Schranke erreicht wird.

Lösungsvorschlag

$$\mathcal{O}(n^2)$$

Diese obere Schranke wird erreicht, wenn die Zahlenfolgen in der umgekehrten Reihenfolge bereits sortiert ist, z. B. 4, 3, 2, 1.

**Die Bschlangaul-Sammlung**

Hermine Bschlangaul and Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.



Diese Materialsammlung unterliegt den Bestimmungen der Creative Commons Namensnennung-Nicht kommerziell-Share Alike 4.0 International-Lizenz.

Hilf mit! Die Hermine schafft das nicht allein! Das ist ein Community-Projekt! Verbesserungsvorschläge, Fehlerkorrekturen, weitere Lösungen sind herzlich willkommen - egal wie - per Pull-Request oder per E-Mail an hermine.bschlangaul@gmx.net. Der TeX-Quelltext dieses Dokuments kann unter folgender URL aufgerufen werden: <https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Staatsexamen/46115/2017/03/Thema-2/Aufgabe-4.tex>