

Einzelprüfung „Theoretische Informatik / Algorithmen (vertieft)“

Einzelprüfungsnummer 66115 / 2019 / Frühjahr

Thema 1 / Aufgabe 6

(Muffinsorten)

Stichwörter: Greedy-Algorithmus

Aus dem Känguru-Wettbewerb 2017 — Klassenstufen 3 und 4.

Luna hat für den Kuchenbasar Muffins mitgebracht: 10 Apfelmuffins, 18 Nussmuffins, 12 Schokomuffins und 9 Blaubeermuffins. Sie nimmt immer 3 verschiedene Muffins und legt sie auf einen Teller. Welches ist die kleinste Zahl von Muffins, die dabei übrig bleiben können?

A: 1, B: 3, C: 4, D: 7, E: 8

- (a) Geben Sie die richtige Antwort auf die im Känguru-Wettbewerb gestellte Frage und begründen Sie sie.

Lösungsvorschlag

4^a

^a<https://www.youtube.com/watch?v=ceJW9kAp1VY>

- (b) Lunas Freundin empfiehlt den jeweils nächsten Teller immer aus den drei aktuell häufigsten Muffinsorten zusammenzustellen. Leiten Sie aus dieser Idee einen effizienten Greedy Algorithmus her, der die Fragestellung für beliebige Anzahlen von Muffins löst (nach wie vor soll es nur vier Sorten und je drei pro Teller geben). Skizzieren Sie in geeigneter Form, wie Ihr Algorithmus die Beispielinstantz von oben richtig löst.

Lösungsvorschlag

```
public static int berechneRest4Sorten3ProTeller() {
    int[] muffins = new int[] { 10, 18, 12, 9 };
    int n = muffins.length;
    sortiere(muffins);

    // Wir nehmen uns 3 verschiedene Muffins solange, wie die dritthäufigste
    // Muffinsorte noch Muffins hat.
    while (muffins[n - 3] > 0) {
        muffins[n - 1]--;
        muffins[n - 2]--;
        muffins[n - 3]--;
        sortiere(muffins);
    }
    return berechneGesamtzahl(muffins);
}
```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/examen/examen_66115/jahr_2019/fruehjahr/Muffin.java](https://github.com/bschlangaul/examen_66115_jahr_2019_fruehjahr/Muffin.java)

- (c) Beschreiben Sie eine mögliche und sinnvolle Verallgemeinerung Ihrer Lösung auf n Muffinsorten und k Muffins pro Teller für $n > 4$ und $k > 3$.

```
public static int berechneRestAllgemein(int[] muffins, int k) {
    int n = muffins.length;
    sortiere(muffins);
    while (muffins[n - k] > 0) {
        for (int i = 1; i <= k; i++) {
            muffins[n - i]--;
        }
        sortiere(muffins);
    }
    return berechneGesamtzahl(muffins);
}
```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/examen/examen_66115/jahr_2019/fruehjahr/Muffin.java](https://github.com/bschlangaul/examen/examen_66115/jahr_2019/fruehjahr/Muffin.java)

- (d) Diskutieren Sie, wie man die Korrektheit des Greedy-Algorithmus zeigen könnte, also dass er tatsächlich immer eine optimale Lösung findet. Ein kompletter, rigoroser Beweis ist nicht verlangt.

Additum

Der komplette Code:

```
import java.util.Arrays;

public class Muffin {

    /**
     * Diese Hilfemethode berechnet die Gesamtzahl der Muffins.
     *
     * @param muffins Ein Feld mit der Gesamtzahl der Muffins je Sorte.
     *
     * @return Gesamtzahl an Muffins (Alle Muffins egal von welcher Sorte addiert)
     */
    public static int berechneGesamtzahl(int[] muffins) {
        int gesamtzahl = 0;
        for (int i = 0; i < muffins.length; i++) {
            gesamtzahl += muffins[i];
        }
        return gesamtzahl;
    }

    /**
     * Sortiere die Anzahl je Muffinsorte aufsteigend. Das letzte Element im Feld
     * ist die Sorte mit den meisten Muffins. Diese Methode gibt das Feld nach der
     * Sortierung auf der Konsole aus. So können wir den Greedy-Algorithmus besser
     * nachvollziehen.
     *
     * @param muffins Ein Feld mit der Gesamtzahl der Muffins je Sorte.
     */
    public static void sortiere(int[] muffins) {
```

```
Arrays.sort(muffins);
System.out.println(Arrays.toString(muffins));
}

/**
 * Implementiert den Greedy-Algorithmus für 4 Muffinsorten und 3 Muffinentnahmen
 * pro Teller.
 *
 * @return Die Anzahl der übrig gebliebenen Muffins.
 */
public static int berechneRest4Sorten3ProTeller() {
    int[] muffins = new int[] { 10, 18, 12, 9 };
    int n = muffins.length;
    sortiere(muffins);

    // Wir nehmen uns 3 verschiedene Muffins solange, wie die dritthäufigste
    // Muffinsorte noch Muffins hat.
    while (muffins[n - 3] > 0) {
        muffins[n - 1]--;
        muffins[n - 2]--;
        muffins[n - 3]--;
        sortiere(muffins);
    }
    return berechneGesamtzahl(muffins);
}

/**
 * Implementiert den Greedy-Algorithmus für allgemeine Werte.
 *
 * @param muffins Ein Feld mit der Gesamtzahl der Muffins je Sorte.
 * @param k        Die Anzahl an verschiedenen Muffins, die pro Teller entnommen
 *                  werden.
 *
 * @return Die Anzahl der übrig gebliebenen Muffins.
 */
public static int berechneRestAllgemein(int[] muffins, int k) {
    int n = muffins.length;
    sortiere(muffins);
    while (muffins[n - k] > 0) {
        for (int i = 1; i <= k; i++) {
            muffins[n - i]--;
        }
        sortiere(muffins);
    }
    return berechneGesamtzahl(muffins);
}

public static void main(String[] args) {
    System.out.println(berechneRest4Sorten3ProTeller());
    System.out.println(berechneRestAllgemein(new int[] { 10, 18, 12, 9, 11 }, 4));
}
}
```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/examen/examen_66115/jahr_2019/fruehjahr/Muffin.java](https://github.com/bschlangaul/examen/examen_66115/jahr_2019/fruehjahr/Muffin.java)



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/66115/2019/03/Thema-1/Aufgabe-6.tex>