Informatik II

Übung 4

Andreas Bärtschi, Andreea Ciuprina, Felix Friedrich, Patrick Gruntz, Hermann Lehner, Max Rossmannek, Chris Wendler

FS 2018

Heutiges Programm

1 Feedback letzte Übung

2 Wiederholung Vorlesung

3 Zur aktuellen Übung

Summen

Gegeben Datenvektor y der Länge $n \in \mathbb{N}$: $(y_i)_{i=1\ldots n} \in \mathbb{R}^n$

Summe $m_n := \sum_{i=1}^{n} y_i$

Summe Quadrate $s_n := \sum_{i=1}^{n} y_i^2$

Neuer Datenpunkt ($n \to n+1$):

$$m_{n+1} = m_n + y_{n+1}$$
$$s_{n+1} = s_n + y_{n+1}^2$$

Provisional Means

Mittelwert $\mu_n := m_n/n$

$$\begin{aligned}
\mu_{n+1} &= \frac{1}{n+1} \sum_{i=1}^{n+1} y_i \\
&= \frac{1}{n+1}(n \cdot \mu_n + y_{n+1}) \\
&= \frac{1}{n+1}((n+1)\mu_n + y_{n+1} - \mu_n) \\
&= \mu_n + \frac{y_{n+1} - \mu_n}{n+1}.
\end{aligned}$$

Mittlere Quadratische Abweichung

$$\sigma_n^2 := \sum_{i=1}^n (y_i - \mu_n)^2 \quad = \sum_{i=1}^n y_i^2 - 2\mu_n y_i + \mu_n^2 = s_n - n \cdot \mu_n^2$$

$$\sigma_{n+1}^2 = s_{n+1} - (n+1) \cdot \mu_{n+1}^2$$

$$= s_n + y_{n+1}^2 - (n+1) \left(\mu_n + \frac{y_{n+1} - \mu_n}{n+1} \right)^2$$

$$= \sigma_n^2 + (y_{n+1} - \mu_n) \cdot (y_{n+1} - \mu_{n+1})$$

Ergebnis

```java
public void addValue(double x){
  ++n;
  if (n == 1){
    mean = x;
    ssq = 0;
  } else {
    double oldMean = mean;
    mean = mean + (x − mean) / n;
    ssq = ssq + (x − oldMean) ∗ (x − mean);
  }
}
```

Beispiel: Lesen vom Scanner

```java
Scanner scanner = new Scanner(System.in);
int number;
while (scanner.hasNextInt()) {
        number = scanner.nextInt();
        ...
}
```

Beispiel: File Lesen

```java
try ( FileReader fr = new FileReader(filename);
      BufferedReader bufr = new BufferedReader(fr);) {
        ...
        String line;
        // read line by line
        while ((line = br.readLine()) != null){
              // do something with line
        }
        ...
} catch (IOException e){
        // do some recovery handling
}
```

Generisches Sortieren mit Java

```java
public class Fruit {
        private String name;        // Attributes
        private int taste;
        private int calories;
        Fruit(String name, int taste, int calories) {
                this.name = name;      // Constructor
                this.taste = taste;
                this.calories = calories;
        }
        public String getName() {    // Name Getter
                return name;
        }
        public int getTaste() {      // Taste Getter
                return taste;
        ...
```

Generisches Sortieren mit Java

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Collections;
public class Main {
        public static void main(String[] args) {
                List<Fruit> fruits = new ArrayList<Fruit>();
                fruits.add(new Fruit("apple", 5, 52));
                fruits.add(new Fruit("pear", 5, 57));
                fruits.add(new Fruit("banana", 6, 89));
                fruits.add(new Fruit("kiwi", 6, 61));
                Collections.sort(fruits);
                for (Fruit fruit : fruits)
                        System.out.println(fruit.getName());
        }
}
```

Generisches Sortieren mit Java

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Collections;
public class Main {
        public static void main(String[] args) {
                List<Fruit> fruits = new ArrayList<Fruit>();
                fruits.add(new Fruit("apple", 5, 52));
                fruits.add(new Fruit("pear", 5, 57));
                fruits.add(new Fruit("banana", 6, 89));
                fruits.add(new Fruit("kiwi", 6, 61));
                Collections.sort(fruits);
                for (Fruit fruit : fruits)
                        System.out.println(fruit.getName());
        }
}
```

Wie sortieren? ⟶

Generisches Sortieren: Comparable Interface

```java
public class Fruit implements Comparable<Fruit> {
        private String name;          // Attributes
        private int taste;
        private int calories;
        Fruit(String name, int taste, int calories) {
                this.name = name;     // Constructor
                this.taste = taste;
                this.calories = calories;
        }
        public String getName() {     // Name Getter
                return name;
        }
        public int getTaste() {       // Taste Getter
                return taste;
        ...
```

Generisches Sortieren: Comparison Method

```java
// Comparison Method for Comparable Interface:
public int compareTo(Fruit that) {
        int BEFORE = -1; int EQUAL = 0; int AFTER = 1;
        // Check if same fruit
        if (this == that)
                return EQUAL;
        // DESCENDING by Taste
        if (this.getTaste() > that.getTaste())
                return BEFORE;
        else if (this.getTaste() < that.getTaste())
                return AFTER;
        else    // Same Taste -> ASCENDING by Calories
                return this.getCalories()-that.getCalories();
}
```

Generisches Sortieren: Comparison Method

```java
        // Comparison Method for Comparable Interface:
        public int compareTo(Fruit that) {
            int BEFORE = -1; int EQUAL = 0; int AFTER = 1;
            // Check if same fruit
            if (this == that)
                    return EQUAL;
            // DESCENDING by Taste
            if (this.getTaste() > that.getTaste())
                    return BEFORE;
            else if (this.getTaste() < that.getTaste())
                    return AFTER;
            else    // Same Taste -> ASCENDING by Calories
                    return this.getCalories()-that.getCalories();
        }
```

negative Zahl
⇔ *"this < that"*

positive Zahl
⇔ *"this > that"*

Fragen oder Anregungen?