

# Programmieren und Software-Engineering I

## Übung 20

Name: \_\_\_\_\_ Klasse: \_\_\_\_\_ Datum: \_\_\_\_\_

### Lernziele:

- Arrays

### Aufgabe 1: Palindrom (Ex\_20\_01\_Palindrom)

Ein **Palindrom** ist ein Wort, das vorwärts und rückwärts gelesen, dasselbe ergibt (abgesehen von Groß- und Kleinschreibung). Erstelle ein Java-Programm, welches prüft, ob es sich bei einem Ausdruck um ein Palindrom handelt. Groß- und Kleinschreibung soll dabei vernachlässigt werden (z.B. mit Hilfe der String-Methode `toUpperCase()` ).

Beispiel:      `String s = "Ebbe".toUpperCase();`      // liefert "EBBE"  
                  `char[] ar = s.toCharArray();`      // liefert char-Array mit {'E','B','B','E'}

Palindrome:      Uhu, neben, Ebbe, Renner, nennen, Rotor, Lagerregal, Rentner, Otto, Anna, Hannah

### Aufgabe 2: Josephus-Problem (Ex\_20\_02\_Josephus)

In einem Kreis stehen  $n$  Personen. Die Personen sind von 1 bis  $n$  durchnummeriert. Nun wird jede  $p$ -te Person aus dem Kreis entfernt. Die übrigen behalten ihre anfänglich zugewiesene Nummer bei. ( $n$  und  $p$  erfolgt durch Benutzereingabe)

Gesucht sind die Nummern der entfernten Personen in der Reihenfolge, in der sie entfernt wurden. Diese Reihenfolge wird Josephus-Permutation genannt (abhängig von  $n$  und  $p$ ).

Beispiel:  
Anzahl der Personen ( $n$ ):      8  
Streichungen ( $p$ ):      5  
  
Entfernung der Personen:      5 – 2 – 8 – 7 – 1 – 4 – 6 – 3

### Aufgabe 3: Geburtstag (Ex\_20\_03\_Birthday)

Erstelle ein Java-Programm, welches folgenden „mathematischen Versuch“ simuliert. Es ist für unterschiedliche Schülerzahlen (10, 20, 23, 29, 30, 40, 50, 60, 70, 80, 90, 100) die Wahrscheinlichkeit zu berechnen, dass zumindest 2 Schüler am selben Tag Geburtstag haben. Um realistische Werte zu erhalten, soll das Experiment mehrmalig (z.B. 10000x) wiederholt werden.

Anschließend soll das Ergebnis entsprechend ausgegeben werden.

Beispiel:      **Anzahl der Versuche:    10000**

	<b>Schülerzahl</b>	<b>Anzahl doppelt</b>	<b>Wahrscheinlichkeit</b>
	<b>10</b>	<b>1151 / 10000</b>	<b>11.51%</b>
	<b>20</b>	<b>4117 / 10000</b>	<b>41.17%</b>
	<b>23</b>	<b>5128 / 10000</b>	<b>51.28%</b>
	<b>29</b>	<b>6870 / 10000</b>	<b>68.70%</b>
	<b>30</b>	<b>...</b>	<b>...</b>
	<b>...</b>		
<b>100</b>	<b>10000 / 10000</b>	<b>100.00%</b>	

# Programmieren und Software-Engineering I Übung 20

### Aufgabe 4: Schneckenrennen (Ex\_20\_04\_SnailRace)

Erstelle ein Java-Programm um das Spiel „Schneckenrennen“ zu simulieren:

Zu Beginn stehen 4 Schnecken am Start. Jeder Spieler würfelt reihum und muss dann mit seiner Schnecke ziehen.

Dabei gelten folgende Regeln für das Würfelergebnis:

- Würfelt er eine 1, 2, 4 oder 5 so fährt er die Anzahl der Felder nach vorne.
- Würfelt er eine 3 so muss er 3 Felder zurück (maximal aber bis zum Start zurück).
- Würfelt er eine 6 so darf er nochmals würfeln und die gesamte Anzahl nach vorne (beim zweiten Wurf ist die 3er-Regel nicht gültig und der Spieler darf 1-6 Felder zusätzlich nach vorne ziehen!!!).

Der Spielplan hat 32 Felder. Das Programm soll ermitteln, welche Schnecke am schnellsten ist und wie viele Runden sie braucht. Achte auf eine benutzerfreundliche Ausgabe.

[illegible]

WICHTIG

Alle Programme müssen einen Programmkopf (=Beschreibung) enthalten.

z.B.:

\*\*\*\*\*

```
*      Name:      Max Mustermann
*      Hü:        4
*      Bsp:        2
*      Datum:      20.10.2015
*      Dateiname:  HUE_04_02_Dreieck.java
*      Beschreibung: Es wird für gegebene Seitenlängen a, b und c geprüft, ob es ein gleichseitiges, ein
                    gleichschenkeliges, ein rechtwinkeliges, ein sonstiges gültiges oder ein ungültiges
                    Dreieck ist.
```

\*\*\*\*\*/

```
public class HUE_04_02_Dreieck {
```