13.6 Übungsblatt 6

Dieses Blatt vertieft die Kenntnisse des Programmverstehens und des Programmierens mit mehrdimensionalen Arrays.

Aufgabe 6.1: ***Lesen***

Lesen Sie sich zunächst die folgenden Kapitel im [Javabuch] durch:

- 6.5 Array-Typ
- 9.2 Methodendefinition und -aufruf

Sie sollten Fragen zu diesen Inhalten beantworten können.

13.6.1 Statistik mit Arrays

Aufgabe 6.2: ***Statistiken für Arrays berechnen***

Aus der Vorlesung sind Arrays als eine Reihung fester Länge bekannt. Sie können verschiedene Werte eines Typs enthalten, sind also vielfältig einsetzbar. Wir betreiben heute etwas Statistik damit. Erstellen Sie bitte eine neue Klasse `ArrayStatistik` mit einer `main`-Methode. Übertragen Sie dazu folgende Liste in ein geeignetes Array:

```
35, 7,  90, 26, 92, 54, 24, 65,  2, 67, 43, 26, 80, 92, 7, 40, 67,
66, 31, 45, 7, 100, 96, 93, 12, 20, 57, 22, 62, 51
```

Bestimmen Sie bitte für obenstehende Werte nun folgende Kennzahlen:

1. Arithmetischer Mittelwert
2. Minimaler Wert und maximaler Wert
3. Median (Tipp: Arrays.sort() sortiert ein Array aufsteigend)

Geben Sie diese Werte bitte am Bildschirm aus.

Hinweis: Recherchieren Sie im Internet oder entsprechender Fachliteratur nach den Berechnungsvorschriften oben gefragter Werte. Berechnen Sie die benötigten Werte vorher beispielhaft mit Stift und Papier. Wie sind Sie dabei vorgegangen? Übertragen Sie Ihre Vorgehensweise in entsprechenden Java-Code.

Testen Sie bitte auch Randfälle ab: Nur ein Element im Array, lauter gleiche Werte im Array, negative Werte im Array, nur negative Werte, sortiertes Array, ...

Aufgabe 6.3: ***Statistik mit mehrdimensionalem Array***

Entstandener Schaden in EUR	Eintrittswahrscheinlichkeit in %
0	74,4
350	18,2
1500	6,1
5000	1,3

Arrays können auch mehrere Dimensionen haben. Die obige Tabelle stellt die Wahrscheinlichkeit für Schadensfälle einer KFZ-Versicherung pro Person innerhalb eines Jahres dar.

Übertragen Sie die Tabelle in ein zweidimensionales Array und berechnen Sie den Erwartungswert des Schadens. Geben Sie den Wert bitte am Bildschirm aus.

13.6.2 Sudoku: Code-Verstehen, Programmieren mit Arrays und Booleans

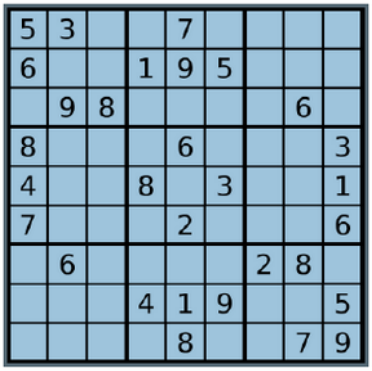


Bild 13.1: Beispiel für ein Sudoku-Spielfeld.

Beim Sudoku wird ein 9x9-Spielfeld mit einigen Zahlen vorgegeben, wobei in jeder Zeile, jeder Spalte und in jedem 3x3-Block die Zahlen 1 bis 9 jeweils genau einmal enthalten sein müssen, siehe auch https://de.wikipedia.org/wiki/Sudoku.

Dazu ist ein Programmrumpf in der Datei `SudokuChecker.java` in ILIAS gegeben.

Aufgabe 6.4: ***Programm installieren und studieren***

- Laden Sie bitte `SudokuChecker.java` herunter und bringen Sie es zum Laufen.
- Beantworten/klären Sie folgende Fragen:
    1. Was sind *erlaubte* Werte im Spielfeld?
    2. Welche Bedeutung hat die "0" dabei?
    3. Warum ist die Reihung `testSpielFelder` dreidimensional?
    4. Was steht in der 1., 2. und 3. Dimension?
    5. Welche Dimension hat die Reihung `testFeld` in der Methode `testSudokuChecker`?
    6. In der Methode `testSudokuChecker` wird `this.spielFeld[8][8]` auf 8 geändert. Ändert sich dadurch `testSpielFelder` auch? Wieso (nicht)?
    7. Wie/wieso funktioniert die For-Each-Schleife in der Methode `testSudokuChecker`? Was wird hier durchlaufen?

Aufgabe 6.5 Teamarbeit: ***Prüfmethoden schreiben***

Die folgende Aufgabe soll in Teams aus drei Personen bearbeitet werden. Finden Sie sich hierzu bitte in dem Team zusammen, dem Sie im ILIAS angehören.

Ziel ist es ein gemeinsames Programm zu entwickeln. Jedes Teammitglied (TM) bekommt hierfür eine der Aufgaben, die mit „TM“ gekennzeichnet sind, zugeteilt.

Bitte kommentieren Sie in Ihrem gemeinsamen Code, wer für welchen Teil zuständig ist.

Vorgehensweise für die Teamarbeit:

- *Präsentieren* Sie sich gegenseitig Ihren Code und klären Sie Fragen, sodass jeder von Ihnen ein Verständnis für den gesamten Code bekommt.
- Geben Sie sich anschließend *Feedback* (Was wurde gut umgesetzt? Was könnte man noch besser machen?). Begründen und diskutieren Sie Ihr Feedback, um voneinander zu lernen.
- *Überarbeiten* Sie Ihren Code gemeinsam, um dadurch Ihre Codequalität zu sichern und achten Sie dabei auch auf eine ausreichende Kommentierung.

a) Alle: Studieren Sie bitte die Methode `validiereSpielfeld` genauer. Dort werden die Methoden

```
isValueOk(wert),
isZeileOk(iZeile, wert),
isSpalteOk(iSpalte, wert),
isBlockOk(iZeile, iSpalte, wert),
```

für jedes Feld im Spielfeld aufgerufen. Die Ausgabe sieht anfangs so aus:

```
...
 +--------+--------+--------+
 | _ _ _  | 5 4 2  | _ _ _  |
 | 9 6 7  | _ _ _  | _ _ _  |
 | _ _ _  | _ _ _  | 3 1 8  |
 +--------+--------+--------+
 | _ _ _  | _ 7 _  | 8 6 4  |
 | _ 2 _  | 6 _ 4  | _ 9 _  |
 | 6 4 5  | _ 1 _  | _ _ _  |
 +--------+--------+--------+
 | 8 9 1  | _ _ _  | _ _ _  |
 | _ _ _  | _ _ _  | 5 4 7  |
 | _ _ _  | 3 2 6  | _ _ 8  |
 +--------+--------+--------+
Womoeglich doppelter Wert auf [8][8] ...
```

Es werden also keine Fehler gefunden, obwohl die 8 in Spalte 9 doppelt ist.

b) Alle: Implementieren Sie bitte die Methode `isValueOk`.

Folgendes gilt auch für die Aufgabenteile c) bis e):

Testen Sie nach jeder erstellten Methode diese ausführlich. Dazu können/sollen Sie natürlich die Methode `testSudokuChecker` zwischenzeitlich umbauen und so auf den jeweiligen Testfall anpassen.

c) TM1: Implementieren Sie bitte die Methode `isZeileOk`.

d) TM2: Implementieren Sie bitte die Methode `isSpalteOk`.

e) TM3: Implementieren Sie bitte die Methode `isBlockOk`. (Hinweis s. unten)

Am Ende soll die Ausgabe mit der ursprünglichen Methode `testSudokuChecker` etwa so aussehen:

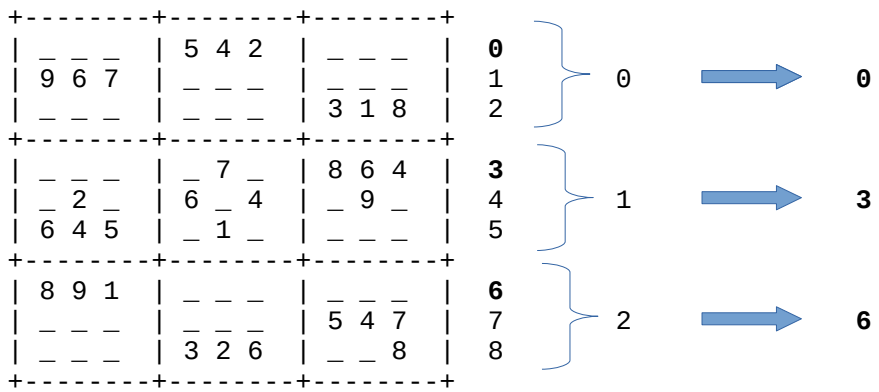
```
 ...
 +--------+--------+--------+
 | _ _ _  | 5 4 2  | _ _ _  |
 | 9 6 7  | _ _ _  | _ _ _  |
 | _ _ _  | _ _ _  | 3 1 8  |
 +--------+--------+--------+
 | _ _ _  | _ 7 _  | 8 6 4  |
 | _ 2 _  | 6 _ 4  | _ 9 _  |
 | 6 4 5  | _ 1 _  | _ _ _  |
 +--------+--------+--------+
 | 8 9 1  | _ _ _  | _ _ _  |
 | _ _ _  | _ _ _  | 5 4 7  |
 | _ _ _  | 3 2 6  | _ _ 8  |
 +--------+--------+--------+
Womoeglich doppelter Wert auf [8][8] ...
Doppelter Spaltenwert 8 in Zeile 3, Spalte 9!
Doppelter Spaltenwert 8 in Zeile 9, Spalte 9!
```

Hinweis zur Methode `isBlockOk`:

Wenn man eine `int`-Variable x mit demselben Wert `t` teilt und wieder mit t malnimmt, kommt meist nicht wieder x heraus: `int x = 8, t = 3; x = x/t*t; // x == 6`
Wie könnte Ihnen dies helfen, um die Blöcke zu identifizieren?

Die folgende Skizze zeigt Ihnen die Zeilenindizes am rechten Rand des Spielfeldes. Mit welcher mathematischen Operation können Sie die Zeilenindizes eines Blocks auf einen gemeinsamen Wert abbilden?

Wie lässt sich aus diesem Wert jeweils der Zeilenindex des ersten Felds (links oben) im jeweiligen Block identifizieren?



Wenn Sie die Formel gefunden haben, stellen Sie die äquivalente Überlegung zur Ermittlung des ersten Spaltenindex innerhalb eines Blocks an.