Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thring

Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik
Wintersemester 2018/19

Objektorientierte Programmierung: Hausaufgabenblatt 10

Abgabe: 14.01.2019, 11:00

Mit diesem Übungsblatt soll das Implementieren eigener Klassen geübt werden.

Hausaufgabe 1: (12 Punkte)

Bitte beachten Sie: die Einsendung der Aufgabe erfolgt nur im YAPEX.
Open exercises via code → Freigabecode: `5d0fa5ba05ea-02c3` (Sudoku)

Ihre Aufgabe ist es, ein Sudokuspiel zu programmieren. Sie müssen dazu eine Klasse `Sudoku` erstellen. Diese Klasse verwaltet das Spielfeld der Größe 9×9 in Form eines zweidimensionalen Arrays vom Typ `byte` und besitzt folgende Funktionen:

- einen Konstruktor, welcher als Parameter das Spielfeld und die Anzahl der bereits enthaltenen Werte erhält. Das Spielfeld ist ein Array vom Typ `byte[][]`, d.h. die Elemente des äußeren Arrays sind selbst Arrays vom Typ `byte[]`.

- eine `print`-Methode, welche keine Parameter und keinen Rückgabewert hat, aber das gesamte Spielfeld ausgibt. Für jede Stelle im Spielfeld wird auf der Konsole entweder die Ziffer, oder, falls diese nicht bestimmt wurde, ein `*` ausgegeben. Es sollten des Weiteren die einzelnen Quadranten erkennbar sein (durch Nutzung der Zeichen `|` und `-`) und die Zeilen- und Spaltennummerierung angegeben werden. Orientieren Sie sich am gegebenen Beispiel.

- eine `setze`-Methode, welche als Parameter die Zeile, die Spalte und den neuen Wert erhält. Der Rückgabetyp ist `char`. Hiermit werden mögliche Fehler codiert:

  'p' falls die **P**osition des einzufügenden Wertes nicht korrekt war (es gibt nur die Zeilen bzw. Spalten 1 bis 9).

  'w' falls der eingegeben **W**ert nicht zwischen 1 und 9 lag.

  'z' falls in der **Z**eile, wo der Wert eingefügt werden soll, bereits solch ein Wert vorkommt.

  's' falls in der **S**palte, wo der Wert eingefügt werden soll, bereits solch ein Wert vorkommt.

  'q' falls in dem **Q**uadranten (dem 3×3 Teilfeld) dieser Wert bereits vorkommt.

  'f' falls der Wert eingefügt wurde und das gesamte Rätsel gelöst wurde und somit **f**ertig ist.

  'l' falls der Wert eingefügt wurde (der Zug **l**egitim war), aber das Rätsel noch nicht gelöst wurde.

  Diese Methode muss somit insbesondere diese Fälle überprüfen, ggf. das Spielfeld entsprechend anpassen und den passenden Wert zurückgegeben. Es ist explizit erlaubt Hilfsmethoden zu erstellen, die einzelne Überprüfungen vornehmen.

- eine `widerrufen`-Methode, welche keine Parameter und keinen Rückgabewert hat. Diese Methode soll immer den letzten Zug zurücknehmen. Es kann immer nur der letzte Zug zurückgenommen werden, d. h. falls ein Wert im Spielfeld geändert wurde, kann diese Änderung rückgängig gemacht werden, aber nicht die Änderung davor.

Das Spiel ist so implementiert, dass es zu Beginn einer jeden Runde das Spielfeld ausgibt. Wurde das Spielfeld in der direkt vorangegangenen Runde geändert, so wird der Spieler gefragt, ob der letzte Zug zurückgenommen werden soll. Anschlieend wird der Nutzer nach der Stelle (Zeile, Spalte) gefragt, welche er in diesem Zug ändern möchte. Direkt folgend, muss der Wert, welcher auf diese Position gesetzt werden soll, eingegeben werden. Falls die setze-Methode der Klasse Sudoku einen Fehler liefert, so wird eine entsprechende Fehlermeldungen ausgegeben. Das Spiel endet mit der Meldung Gratuliere, Sie haben das Raetsel geloest!, wenn Stellen im Spielfeld belegt sind.

Das Hauptprogramm (in der Klasse `TestSudoku`) ist bereits vorgegeben. Sie müssen nur die geforderte Klasse zur Verwaltung des Spielfeldes schreiben. Insbesondere wird die Benutzer-Interaktion schon vom vorgegebenen Hauptprogramm erledigt.

```
$ java TestSudoku
  123 456 789
  --- --- ---
1|***|9**|728|
2|278|**3|*1*|
3|*9*|***|64*|
  --- --- ---
4|*5*|*6*|2**|
5|**6|***|3**|
6|*1*|*5*|***|
  --- --- ---
7|1**|7*6|*34|
8|***|5*4|***|
9|7*9|1**|8*5|
Zeile:  12
Spalte: 7
Wert:   5
Wert muss zwischen 1
   und 9 liegen!
Zeile:  2
Spalte: 7
Wert:   50
Wert muss zwischen 1
   und 9 liegen!
Zeile:  2
Spalte: 7
Wert:   5
  123 456 789
  --- --- ---
1|***|9**|728|
2|278|**3|51*|
3|*9*|***|64*|
  --- --- ---
4|*5*|*6*|2**|
5|**6|***|3**|
6|*1*|*5*|***|
  --- --- ---
7|1**|7*6|*34|
8|***|5*4|***|
9|7*9|1**|8*5|
Letzten Zug
   widerrufen? (j
   fuer ja, n fuer
   nein) n
Zeile:  3
Spalte: 9
Wert:   1
1 bereits im Quadrant
    enthalten!
```

```
Zeile:  2
Spalte: 5
Wert:   1
1 bereits in Zeile
   enthalten!
Zeile:  1
Spalte: 1
Wert:   1
1 bereits in Spalte
   enthalten!
Zeile:  5
Spalte: 8
Wert:   5
  123 456 789
  --- --- ---
1|***|9**|728|
2|278|**3|51*|
3|*9*|***|64*|
  --- --- ---
4|*5*|*6*|2**|
5|**6|***|35*|
6|*1*|*5*|***|
  --- --- ---
7|1**|7*6|*34|
8|***|5*4|***|
9|7*9|1**|8*5|
Letzten Zug
   widerrufen? (j
   fuer ja, n fuer
   nein) n
Zeile:  3
Spalte: 3
Wert:   1
  123 456 789
  --- --- ---
1|***|9**|728|
2|278|**3|51*|
3|*91|***|64*|
  --- --- ---
4|*5*|*6*|2**|
5|**6|***|35*|
6|*1*|*5*|***|
  --- --- ---
7|1**|7*6|*34|
8|***|5*4|***|
9|7*9|1**|8*5|
Letzten Zug
   widerrufen? (j
   fuer ja, n fuer
   nein) j
```

```
  123 456 789
  --- --- ---
1|***|9**|728|
2|278|**3|51*|
3|*9*|***|64*|
  --- --- ---
4|*5*|*6*|2**|
5|**6|***|35*|
6|*1*|*5*|***|
  --- --- ---
7|1**|7*6|*34|
8|***|5*4|***|
9|7*9|1**|8*5|
```

weitere Ein- und Ausgaben

```
123 456 789
  --- --- ---
1|634|915|728|
2|278|643|519|
3|591|278|643|
  --- --- ---
4|457|369|281|
5|986|421|357|
6|312|857|496|
  --- --- ---
7|125|786|934|
8|863|594|172|
9|749|132|8*5|
Zeile:  9
Spalte: 8
Wert:   6
  123 456 789
  --- --- ---
1|634|915|728|
2|278|643|519|
3|591|278|643|
  --- --- ---
4|457|369|281|
5|986|421|357|
6|312|857|496|
  --- --- ---
7|125|786|934|
8|863|594|172|
9|749|132|865|
Gratuliere, Sie haben
    das Raetsel
   geloest!
```