Aufgaben zur Veranstaltung
Programmierung mit Java / IT-Grundlagen, WS 2023/2024

H. Pflug, J. Dietel FH Aachen, Campus Jülich; IT Center, RWTH Aachen

Hausaufgaben 7 15./16.11.2023

Abgabe der Lösung am 21.11.2023

Schiebepuzzle

Im sogenannten "Schiebepuzzle"' sind 15 Plättchen in einem 4x4 großen Rahmen angeordnet. Ein Feld bleibt frei (siehe Bild). Durch horizontales und vertikales Verschieben können die Plättchen gemischt werden. Das Puzzle besteht darin, anschließend die oben abgebildete Reihenfolge wiederherzustellen.
In dieser Aufgabe sollen Sie eine Klasse `Schiebepuzzle` mit den wichtigsten Funktionen schreiben.

Tipp: Lesen Sie sich die Aufgaben komplett durch, bevor Sie beginnen, damit Sie wissen, welche Anforderungen die Klasse erfüllen muss.

Schreiben Sie zunächst die Klasse selbst mit den nötigen Attributen und einen Konstruktor

```
public Schiebepuzzle()
```

der ein Schiebepuzzle erzeugt, in dem die Zahlen richtig (wie in der obigen Abbildung) angeordnet sind.

Schreiben Sie eine Methode

```
public void schiebe(int i)
```

Diese Methode schiebt das Plättchen mit der Nummer `i` auf den freien Platz. Schreiben Sie dazu zwei Exception-Klassen, die beide `RuntimeExceptions` sein sollen:

```
public class WrongNumberException   //i liegt nicht zwischen 1 und 15
public class WrongMoveException     //Zug ist nicht moeglich
```

und werfen Sie an den entsprechenden Stellen die jeweilige Exception. Mit einer weiteren Methode

```
public boolean istVerschiebbar(int i)
```

kann überprüft werden, ob das Plättchen Nummer `i` verschiebbar ist, d.h. neben dem leeren Feld liegt.
Schreiben Sie eine weitere Methode

```
public void mische()
```

Diese Methode mischt das Spiel, indem sie 100 zufällige (gültige) Züge vornimmt.
Anmerkung: Diese Methode garantiert, dass das Puzzle immer lösbar ist. Wenn man die Plättchen einfach zufällig anordnet, ist das nur mit einer Wahrscheinlichkeit von 50 % der Fall.

Schreiben Sie eine `toString()`-Methode, um sich das Puzzle anzeigen zu lassen.

Lösungsalgorithmen

Das Puzzle soll sich auch automatisch lösen lassen. Der Benutzer soll dabei die Wahl zwischen verschiedenen Lösungsalgorithmen haben. Die Lösungsalgorithmen selbst sind noch unbekannt.

Schreiben Sie einen objektorientierten Rahmen, in den die Algorithmen später eingesetzt werden können. Verwenden Sie als Ansatz das Interface

```
public interface Loesungsalgorithmus
```

- Der Lösungsalgorithmus darf das Schiebepuzzle nur durch Aufruf von `schiebe` verändern. Er benötigt Informationen, an welcher Position die Zahlen im Schiebepuzzle stehen. Ergänzen Sie die Klasse `Schiebepuzzle` entsprechend.
- Fügen Sie die benötigten Methode(n) in das Interface ein.
- Schreiben Sie einen ersten einfachen „Lösungs“-algorithmus, um das Interface zu testen:

  ```
  public SchiebAlg1 implements Loesungsalgorithmus
  ```

  Der Algorithmus verschiebt so lange zufällige Plättchen, bis das Plättchen Nummer 1 an seinem Platz steht (links oben in der Ecke). Die restlichen Plättchen werden von diesem Algorithmus nicht sortiert (das würde zu lange dauern).
- Schreiben Sie eine Testklasse, in der ein Schiebepuzzle erzeugt, gemischt und anschließend „gelöst“ wird.

Testfall

```
public static void main(String[] args) {
  Schiebepuzzle puzzle = new Schiebepuzzle();
  // Mischen nicht vergessen, sonst hat der Spieler sehr schnell gewonnen
  puzzle.mische();
  // Testen des Loesungsalgorithmus
  // -> zufaellig schieben
  Loesungsalgorithmus alg1 = new SchiebAlg1();
  alg1.loese(puzzle);
}
```

Beispielhafte Ausgabe

Vor dem Loesen:

```
| 7| 8|11| 2|
-------------
|14|  | 6|15|
-------------
| 1| 4| 9| 3|
-------------
|13| 5|10|12|
-------------
```

Nach dem Loesen:

```
| 1|  | 3| 2|
-------------
|11|12|15|13|
-------------
|10| 9| 6| 5|
-------------
|14| 7| 4| 8|
-------------
```