Portfolioprüfung
Einzelaufgabe

Zimmermann, Himmelspach

Nordakademie

21. September 2018




Allgemeine Randbedingungen

- Sie bearbeiten die Aufgabenstellung einzeln.
- Im Falle einer Mehrfachabgabe des Codes werden alle beteiligten Abgaben mit 0 Punkten bewertet.
- Abgabe bis zum 7. Oktober 2017 24:00 Uhr möglich.
- Ein Feedbacktermin ist nicht mehr vorgesehen.
- Sie erhalten maximal 32 Punkte für diese Programmieraufgabe. Über das Ergebnis dieses Prüfungsteils können sie sich während der Prüfungseinsicht informieren.

Aufgabenstellung Domino Spielbeginn 1/5

1. Es soll eine vereinfachte Variante des Dominospiels für zwei oder mehr Spieler implementiert werden. Dabei werden mehrere Runden gespielt. Lesen Sie die Aufgabenstellung vollständig, bevor Sie mit dem Programmieren anfangen. Diese Beschreibung bezieht sich auf zwei Spieler.
2. Es gibt 25 Spielsteine, auf denen jeweils unterschiedliche Paare ganzer Zahlen stehen. Die Zahlen sind größer-gleich 0 und kleiner-gleich 4.
3. Am Anfang des Spiels werden jeweils die fünf ersten Spielsteine aus einem Pool genommen und an den ersten und danach fünf weitere an den zweiten Spieler verteilt. Dann wird der nächste Spielstein, z.B. [0|4] offen aufgedeckt.
4. Dann können die Spieler abwechselnd Spielsteine links oder rechts anlegen. In jeder Spielrunde (vgl. Punkt 12) beginnt immer der erste Spieler. Dabei dürfen im Beispiel an [0|4] auf der linken Seite nur Steine angelegt werden, die rechts eine 0 stehen haben, z.B. [3|0]. Analoges gilt für die rechte Seite. Ein Drehen der Steine, also aus [3|0] eine [0|3] machen, ist nicht erlaubt.

Aufgabenstellung Domino Spielregeln 2/5

5. Falls ein Spieler keinen Stein anlegen kann, muss er, solange noch restliche Spielsteine zur Verfügung stehen, einen Stein ziehen und zu seinen hinzufügen. Es besteht keine Pflicht anzulegen. Wenn noch Steine vorhanden sind kann man sich stattdessen entscheiden, einen Stein zu ziehen.
6. Steine können immer nur ganz links oder ganz rechts angelegt werden. Liegen die Steine [3|0] [0|4], kann links nur ein Stein angelegt werden, der auf der rechten Seite eine 3 und rechts nur ein Stein, der auf der linken Seite eine 4 stehen hat. (Hinweis: Analytisch begabte Menschen sehen also nicht die Steinkette [3|0] [0|4] sondern nur „einen Stein“ [3|4] an den angelegt werden soll.)

Aufgabenstellung Domino Spieler 3/5

7. Es gibt einen menschlichen Spieler, der über die Konsole gesteuert wird, und zwei Arten von Computerspielern: einen einfältigen, der immer die erste angebotene Variante wählt, und einen hazardierenden, der eine zufällige Variante wählt.
8. Der menschliche Spieler soll immer alle seine Steine und die Steine die er anlegen könnte, angezeigt bekommen. (Siehe Folie 10) Danach kann er einen Stein wählen oder sich entscheiden, einen Stein zu ziehen. Falls der menschliche Spieler nicht anlegen kann, wird automatisch ein Stein gezogen (sichtbar durch Ausgabe „Keine Auswahlmöglichkeit“).
9. Beachten sie, dass es passieren kann, dass an [3|4] angelegt werden soll und der Spieler einen Stein [4|3] hat, der an beiden Seiten passen könnte. In diesem Fall ist der Spieler zu befragen, ob er links oder rechts anlegen möchte.
10. Die Computerspieler sollen den Stein, den sie anlegen ausgeben. Falls ein Computerspieler stattdessen einen Stein zieht, ist dieses auch auszugeben.

Aufgabenstellung Domino Spielende 4/5

11. Das Spiel ist beendet, wenn entweder ein Spieler seinen letzten Stein abgelegt hat oder keine restlichen Spielsteine zum Ziehen zur Verfügung stehen und die Spieler keine Spielsteine mehr anlegen können. Am Ende des Spiels werden alle Zahlen auf den Spielsteinen, die ein Spieler noch hat, als Minuspunkte zusammengezählt.
12. Nachdem ein Spiel beendet ist, wird abgefragt ob eine weitere Runde gespielt werden soll, dabei werden die dann erreichten Minuspunkte zu den vorhandenen addiert und angezeigt.

Aufgabenstellung Domino Sonstiges 5/5

Um eine zügige Korrektur zu gewährleisten beachten sie bitte:

13. Nutzen Sie die Klasse `DominoPool` von Folie 12, die Ihnen einen gemischten Satz von Dominosteinen zur Verfügung stellt. D.h. jedoch nicht, dass Sie zum testen nicht eine abgewandelte Form verwenden dürfen...
14. Nutzen Sie auch die Klasse User Dialog, die in Folie 13f dargestellt ist.
15. Erzeugen Sie eine Klasse Main mit einer Exemplarmethode `startDomino(`. Diese Methode soll alle notwendigen Objekte erzeugen und das Spiel für einen menschlichen (Sie) und einen einfältigen Computerspieler (Ich) starten. Eine Auswahlmöglichkeit, als menschlicher Spieler gegen einen hazardierenden Spieler oder gar gegen einen anderen menschlichen Spieler anzutreten ist an dieser Stelle nicht erwünscht.
16. Bitte beachten Sie die Reihenfolge des Verteilens an die Spieler in Punkt 3!

Ihre Implementierung des Spiels soll unter anderem folgende Klassen enthalten (Aufzählung ist verpflichtend aber nicht unbedingt erschöpfend noch eine gewünschte Musterlösung):

- `DominoGame`: Ein Objekt dieser Klasse steuert das Spiel, ist also für die Erzeugung und Verteilung der Spielsteine und die Durchführung des Spiels verantwortlich.
- `HumanPlayer`: Ein Spielerobjekt das seine Auswahl aus der Liste der möglichen Züge über die Console eingibt.
- `ComputerPlayer`: Ein Computerspielerobjekt, das seine Auswahl aus der Liste der möglichen Züge nach einem Algorithmus ermittelt.
- `Domino`: Objekte dieser Klasse repräsentieren einen Spielstein.

Bewertungskriterien

Rechnen Sie mit einer Bearbeitungszeit von 20 h. Bewertungskriterien für diese Aufgabe sind unter anderem:

- Ihr Programm erfüllt die funktionalen Anforderungen und stürzt nicht mit einer Exception ab.
- Sie halten sich an die Java-Namenskonventionen und verwenden aussagekräftige Namen.
- Methoden sind gemäß besprochener Konventionen dokumentiert.
- Sie vermeiden überflüssigen Code. Versuchen Sie die Aufgabe so einfach wie möglich zu lösen. Versuchen Sie also z.B. nicht, dem Computerspieler Intelligenz einzuhauchen.
- Sie verwenden Collections und Arrays in effizienter Weise.
- Sie nutzen Klassenhierarchien wie in den Vorlesungen dargestellt.

Durch Tests können Sie die Qualität der Software sicherstellen bzw. erhöhen. Die Tests sind jedoch **nicht** Bestandteil der Bewertung.

Beispielablauf 1/2

Der folgende Ausdruck gibt einen Beispielablauf wieder. Vereinfachend wurde mit 0 bis 3 als Werte auf den Dominosteinen gespielt, und auch nur 4 Dominosteine vergeben. Bitte halten Sie sich an den Beispielablauf.

```
Anlegemöglichkeit: [3|0]
Ihre Steine: [[2|3], [0|2], [1|2], [0|0]]
Auswahlmöglichkeiten:
(0) [0|0] (1) [2|3] (2) [0|2] (3) ziehen
0
Anlegemöglichkeit: [3|0]
Ich: [3|3]
Anlegemöglichkeit: [3|0]
Ihre Steine: [[2|3], [0|2], [1|2]]
Auswahlmöglichkeiten:
(0) [2|3] (1) [0|2] (2) ziehen
0
Anlegemöglichkeit: [2|0]
Ich: [0|1]
Anlegemöglichkeit: [2|1]
Ihre Steine: [[0|2], [1|2]]
Auswahlmöglichkeiten:
(0) [1|2] (1) [0|2] (2) ziehen
1
Anlegemöglichkeit: [0|1]
Ich: [1|0]
Ich: links anlegen
Anlegemöglichkeit: [1|1]
Ihre Steine: [[1|2]]
Auswahlmöglichkeiten:
(0) [1|2] (1) ziehen
1
Anlegemöglichkeit: [1|1]
Keine Anlegemöglichkeit
Anlegemöglichkeit: [1|1]
Ihre Steine: [[1|2], [2|1]]
Auswahlmöglichkeiten:
(0) [2|1] (1) [1|2] (2) ziehen
1
Anlegemöglichkeit: [1|2]
Ich: [3|1]
Anlegemöglichkeit: [3|2]
Ihre Steine: [[2|1]]
Auswahlmöglichkeiten:
(0) [2|1] (1) ziehen
0
Spielende
Sie: []
Ich: [[0|3]]
Sie-Ich 0:3
Weitere Runde? (0) Nein (1) Ja
1
```

Beispielablauf 2/2

```
Anlegemöglichkeit: [2|3]
Ihre Steine: [[1|2], [3|0], [0|3], [0|0]]
Auswahlmöglichkeiten:
(0) [1|2] (1) [3|0] (2) ziehen
1
Anlegemöglichkeit: [2|0]
Ich: [0|2]
Ich: links anlegen
Anlegemöglichkeit: [0|0]
Ihre Steine: [[1|2], [0|3], [0|0]]
Auswahlmöglichkeiten:
(0) [0|0] (1) [0|3] (2) ziehen
0
Auswahlmöglichkeiten:
(0) links anlegen (1) rechts anlegen
1
Anlegemöglichkeit: [0|0]
Ich: [2|0]
Anlegemöglichkeit: [2|0]
Ihre Steine: [[1|2], [0|3]]
Auswahlmöglichkeiten:
(0) [1|2] (1) [0|3] (2) ziehen
1
Anlegemöglichkeit: [2|3]
Ich: [3|3]
Anlegemöglichkeit: [2|3]
Ihre Steine: [[1|2]]
Auswahlmöglichkeiten:
(0) [1|2] (1) ziehen
0
Spielende
Sie: []
Ich: [[1|3]]
Sie-Ich 0:7
Weitere Runde? (0) Nein (1) Ja
0
Tschüß
```

Klasse DominoPool (Unbedingt verwenden!)

```
import java.util.Collections;
import java.util.LinkedList;
import java.util.List;
public class DominoPool {
 private static final int MAX_NUMBER_ON_DOMINO = 5;
 public List<Domino> provideShuffledDominoHeap() {
  List<Domino> heapOfDominoes = new LinkedList<>();
  for (int left = 0; left < MAX_NUMBER_ON_DOMINO; left++) {
   for (int right = 0; right < MAX_NUMBER_ON_DOMINO; right++) {
    heapOfDominoes.add(new Domino(left, right));
   }
  }
  Collections.shuffle(heapOfDominoes);
  return heapOfDominoes;
 }
}
```

Klasse mit Userdialog 1/2

```
import java.util.Scanner;
public class UserDialog {
private static Scanner scanner = new Scanner(System.in);
int getUserInput(String label, String... alternatives) {
 do {
  promptUser(label, alternatives);
  int input = scanUserInput();
  if (input < 0 || input >= alternatives.length) {
   System.out.println("Fehler! Eingabe bitte wiederholen.");
  } else {
    return input;
  }
 } while (true);
}
```

Klasse für Userdialog 2/2

```
private int scanUserInput() {
 try {
  String inputString = scanner.nextLine();
  return Integer.parseInt(inputString);
 } catch (NumberFormatException nfe) {
  return -1;
 }
}
private void promptUser(String label, String... alternatives) {
 System.out.print(label);
 int index = 0;
 for (String alternative : alternatives) {
  System.out.format("(%d)␣%s␣", index++, alternative);
 }
 System.out.println();
}}
```

Hochladen der Lösung

1. Speichern Sie ihr Projekt als jar Datei <xxx>_Domino.jar
2. xxx steht dabei für Ihren Nachnamen
3. Laden Sie die Datei in moodle hoch.