Programmierung 1
Wintersemester 2023/24
Prof. Dr.-Ing. Johannes Konert

Hausaufgabe 8

Abgabe Mi, 27.12.2023 (23:59 Uhr)

Abgabe:
Für dieses Hausaufgabenblatt sind **eine** oder **zwei** Dateien abzugeben: ***Article.java***, ***ArticleTest.java*** ODER ***BrainDice.java*** (Sie wählen aus, welche der beiden Aufgaben Ihnen eher gefällt). Jede abgegebene Klasse muss lauffähig sein. Nehmen Sie auch Quellcodebeispiele der SU zur Orientierung.

Aufgabe 0A: Shopsystem mit Artikeln
Implementieren Sie eine Klasse `Article`, deren Instanzen Warenprodukte eines Online-Shops repräsentieren.

1. Ein Artikel sollte mindestens einen Titel, einen Preis und eine eindeutige Artikelnummer haben. *Es gibt Ähnlichkeiten zur Implementierung des Kontos aus der SU, woran Sie sich orientieren können (insbesondere bezüglich der Vergabe der Artikelnummern).*
2. Ergänzen Sie die Klasse `Article` um Klassenattribute und Klassenmethoden für die Anzahl aller verfügbaren Artikel.
3. Implementieren Sie eine Klasse `ArticleTest` mit einer `main`-Methode und instanziieren Sie darin einige Artikel zum Test. Geben Sie die Artikelbezeichnungen, Artikelnummern und Preise sowie Gesamtzahl an Artikeln auf die Konsole aus. *(Tipp: Wenn Sie mögen, implementieren Sie dafür eine Objektmethode* `public String toString()` *für* `Article`*).*
4. Ergänzen Sie in `Article` eine Klassenmethode `public static void changePrices(Article[] articles, double percentage)`, welche alle Artikel des übergeben Arrays um den in percentage angegebenen Prozentsatz im Preis anpasst. *(Denkanstoß: Warum braucht die Methode keine Rückgabe, ist also void?)*
5. *Übung für die Prüfung:* Schreiben Sie auf, welche Attributewerte, Variablen und Objekte sich alle im Metaspace, im Heap und im Stack befinden, wenn Sie am Ende der main-Methode anhalten würden.

Aufgabe 0B: Würfel mit Gedächtnis

1. Schreiben Sie eine Klasse `BrainDice`, welche garantiert nur eine Instanz zulässt (Singleton). Diese Instanz repräsentiert einen Würfel, dessen maximale Augenzahl (entspricht den Würfelseiten) vor der ersten Nutzung bestimmt werden kann (z.B. durch Aufrufen einer Methode zum Setzen der Augenzahl oder bei Instanziierung). Der Würfel soll mindestens eine Methode `public int nextThrow()` haben, welche eine Augenzahl zwischen 1 und der maximalen Augenzahl (beide inklusive) zurückgibt.
   Das Besondere am `BrainDice` ist, dass er/sie sich merkt, welche Zahlen bereits gefallen sind und diese nicht mehr ausgibt, bis alle möglichen Würfelaugen einmal gefallen sind; dann setzt der Würfel sich selbst zurück und liefert erneut die Zahlen aus (in neuer zufälliger Reihenfolge).
   Nutzen Sie für Zufallszahlen
   `Random rnd = new java.util.Random(); int num = rnd.nextInt(maxValue+1);`[1]
   Beispiel: Ein `BrainDice` *mit max. 6 Augen liefert bei mehrmaligem Aufruf von* `nextThrow()` *zunächst die 2, dann 4, dann 1, dann 5. Beim nächsten Aufruf von* `nextThrow()` *kann nur 3 oder 6 als Rückgabewert kommen. Nach sechs Aufrufen von* `nextThrow()` *sind also alle Zahlen genau einmal zurückgegeben worden und der Würfel liefert anschließend beim siebten Aufruf von* `nextThrow()` *wieder eine der Zahlen von 1 bis 6, usw.*
2. Schreiben Sie in der `main`-Methode von `BrainDice` Anweisungen, welche zwei Mal eine Instanz von BrainDice abrufen (und natürlich in beiden Fällen das gleiche Objekt erhält). Dies geht, indem Sie z.B. zwei Mal `.getInstance()` aufrufen. Speichern Sie beide Referenzen in verschiedenen Variablen. Nehmen Sie als Augenzahl z.B. 9 und rufen Sie anschließend doppelt so oft (z.B. 18 mal) die Methode `nextThrow()` auf und geben Sie die Zahlen auf die Konsole aus. Verwenden Sie dabei beide Referenzvariablen (z.B. im Wechsel). Das Verhalten sollte identisch sein, egal ob Sie nur eine oder beide Referenzvariablen verwenden, da die gleiche Instanz des Objektes referenziert wird (Singleton).

(Knobelaufgabe auf der nächsten Seite)

[1] https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Random.html#nextInt(int)

Knobelaufgabe

Wie immer beim Knobeln: Gehen Sie das folgende Programm in Gedanken durch, auch wenn es etwas merkwürdig aussieht und auch die üblichen Namenskonventionen nicht beachtet. Beantworten Sie die folgende Frage: **Gibt es Hinz oder Kunz auf der Konsole aus?**
So viel sei verraten: Das Programm kompiliert und läuft und die Antwort ist eindeutig. **Warum ist das so?**

*Dieser Quellcode vermittelt nicht nur tieferes Verständnis für die Verwendung von Klassen in Klassen (ja auch das geht) sondern auch über die Präzedenzregeln von Klassen und lokalen Variablen. Um es für alle Programmierenden immer leicht zu machen beim Quellcodelesen schreiben wir auch deshalb Klassen am Anfang **G**roß und Variablen **k**lein, so entsteht keine Zweideutigkeit beim Lesen des Quellcodes für den Menschen.*

```
public class HinzOrKunz {
  public static void main(String[] args) {
    System.out.println(Outer.Inner.Name);
  }
}

class Outer {
  static class Inner {
    static String Name = "Hinz";
  }
  static Helper Inner = new Helper();
}

class Helper {
  String Name = "Kunz";
}
```

Lösungs-Erläuterung (weiße Zeilen markieren und in anderem Programm einfügen):

Ende Erläuterung.