hochschule mannheim

FAKULTÄT FÜR INFORMATIK

Prof. Dr. Peter Knauber

PR1 – Programmieren 1 | Bachelor-Studiengang Informatik | WS2019/2020

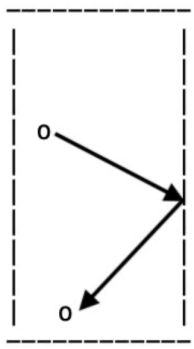
Übungsblatt 8: Klassen und Objekte

Ausgabe: 25.11.19
Abgabe: 11.12.19 10:00 Uhr elektronisch mittels Git

Aufgabe 1

Erstellen Sie ein Java-Programm (Klasse `Playground`) für ein zweidimensionales Spielfeld der Größe 10x10.

Erstellen Sie eine Klasse `Ball` mit der initialen Position (2, 3); der Ball soll sich auf dem Spielfeld bewegen können (Methode `move()`). Er beginnt seine Bewegung mit der Geschwindigkeit ein Feld / Schritt nach oben, sowie ein Schritt nach rechts. Wenn der Ball eine Wand erreicht soll er davon abprallen (siehe Bild). Seine Geschwindigkeit ändert sich dabei nicht. Der Ball darf das Spielfeld nicht verlassen. Nur der Ball kennt seine aktuelle Position, d.h. diese ist nicht von außen sichtbar. Sie dürfen der Klasse `Ball` andere geeignete Methoden geben, aber die Position nicht herausreichen; insbesondere sind keine `get`-Methoden für die Position erlaubt. Immer, wenn der Ball die untere Linie erreicht (und abprallt), soll in der Klasse `Playground` ein Zähler erhöht werden, der angibt, wie häufig die untere Linie getroffen wurde.



Schreiben Sie eine Methode `runGame()` in der Klasse `Playground`, die 1000 Schritte des Balls auslöst und auf der Konsole ausgibt. In jedem Schritt soll der Ball bewegt werden (Methode `move` der Klasse `Ball`) und danach das Spielfeld mit dem Ball an seiner neuen Position ausgegeben werden. Die Ausgabe soll jeweils um 100ms verzögert werden. Verwenden Sie dazu den folgenden Code in der `runGame`-Methode, der an dieser Stelle die Ausführung verzögert:

```
print(field);
try {
       Thread.sleep(100);    // 100ms warten
} catch (InterruptedException e) {
}
```

Wenn Sie nun Ihre Eclipse-Konsole geeignet hoch machen scheint der Ball sich zu bewegen.

Schreiben Sie eine `main`-Methode in der Klasse `Playground`, die das Spielfeld erzeugt und dann das Spiel startet. Geben Sie am Ende aus, wie oft der Ball die untere Linie getroffen hat.

Aufgabe 2

Erstellen Sie eine Klasse `BigInt`, welche das Rechnen mit sehr großen natürlichen Zahlen (inklusive 0) ermöglicht.

Der Konstruktor der Klasse soll eine Zahl als `String` entgegennehmen und dann ziffernweise in einem Feld vom Typ `int[]` speichern. Besteht der Parameter lediglich aus Ziffern, dann wird das Array mit der nötigen Länge erzeugt (es soll lediglich so groß wie unbedingt notwendig sein) und die Zahl wird ziffernweise darin gespeichert. Bei ungültigem Parameter (keine natürliche Zahl im String) soll der Konstruktor eine `PRException` mit geeigneter Fehlermeldung auslösen. Beispiel:

```
BigInt zahl1 = new BigInt("23");
BigInt zahl2 = new BigInt("45");
```

In dem String, der an den Konstruktor übergeben wird, können führende Nullen enthalten sein, z.B. “04”. In den Rückgabewerten der folgenden Methoden dürfen jedoch keine führenden Nullen enthalten sein, z.B. soll der Aufruf `new BigInt("04").toString()` den String “4” liefern. Auch Additionsergebnisse dürfen keine führenden

Nullen enthalten.
Die Zahl “0” muss natürlich korrekt behandelt werden.

Die Klasse `BigInt` soll über folgende Methoden verfügen:

- `void add( BigInt number )`

  Addiert die als Parameter übergebene Zahl vom Typ `BigInt` zu der im Objekt vorliegenden Zahl hinzu. Das Ergebnis der Addition wird also anstelle der zuvor vorliegenden Zahl im Objekt gespeichert (so dass man damit weiterarbeiten kann). Das als Parameter übergebene Objekt bleibt unverändert. Beispiel:

  ```
  zahl1.add(zahl2);     // die Variable zahl2 wurde oben vereinbart
  ```

  Tipp: Vergegenwärtigen Sie sich bei der Implementierung die schriftliche Addition.

- `String toString()`

  Liefert die vorliegende Zahl als String. Beispiel:

  ```
  println(zahl1.toString());   // gibt "68" auf der Konsole aus
  println(zahl2.toString());   // gibt "45" auf der Konsole aus
  ```

- `int[] getDigits()`

  Liefert das Feld in dem die Ziffern der vorliegenden Zahl gespeichert wurden. Beispiel:

  ```
  println(zahl1.getDigits()[0]); // gibt "6" auf der Konsole aus
  println(zahl1.getDigits()[1]); // gibt "8" auf der Konsole aus
  ```

- `int length()`

  Liefert die Anzahl der Ziffern der vorliegenden Zahl. Beispiel:

  ```
  println(zahl1.length());            // gibt "2" auf der Konsole aus
  ```

- `boolean equals(BigInt number)`

  Liefert genau dann `true`, wenn beide Objekte die gleiche Zahl repräsentieren, sonst `false`. Beispiel:

  ```
  println(zahl1.equals(zahl2));              // gibt "false" auf der Konsole aus
  println(zahl1.equals(new BigInt("68")));   // gibt "true" auf der Konsole aus
  ```

- `boolean greaterThan(BigInt number)`

  Liefert genau dann `true`, wenn die vorliegende Zahl größer ist als der Parameter `number`, sonst `false`. Beispiel:

  ```
  println(zahl1.greaterThan(zahl2));  // gibt "false" auf der Konsole aus
  println(zahl2.greaterThan(zahl1));  // gibt "true" auf der Konsole aus
  ```

a) Erstellen Sie die Klasse `BigInt` wie oben beschrieben.
b) Erstellen Sie in `BigInt` eine `main`-Methode, um Ihre Methoden zu testen.

Allgemeines

- Legen Sie für die Bearbeitung dieses Übungsblattes ein Paket namens `uebung08` an, in dem Sie Ihre Klassen anlegen.
- In Moodle finden Sie Testklassen `...1stTest`, mit der Sie prüfen können, ob Ihre Methoden korrekt definiert sind und zumindest die einfachsten Eingaben korrekt behandelt werden.
- Erlaubt sind `MakeItSimple`-Funktionen (keine nicht besprochene Funktionalität aus der Java-Standard-Bibliothek) und das bisher erworbene Wissen aus den PR1-Vorlesungen. Sie müssen alle(!) von Ihnen verwendeten Konstrukte der Sprache sowie alle verwendeten Methoden, die nicht aus der Hilfsbibliothek `MakeItSimple` stammen, gut erklären und nötigenfalls im Testat selbst programmieren können!
- Sie geben ab, indem Sie vor Ende der Abgabefrist Ihr Projekt mit den lauffähigen Programmen des Übungsblattes in das Repository auf *wilma* pushen. Achten Sie darauf, ob Sie die Kontroll-E-Mail bekommen! Die letzte hochgeladene Version Ihres Projekts wird gewertet. Andere Abgaben, ob elektronisch oder auf Papier, zählen als **nicht abgegeben**!
- Die Aufgaben sind in Eclipse zu bearbeiten und beim Testat vorzuführen.
- Nutzen Sie die Übungsstunde, um eventuelle Fragen zur Vorlesung, zur Übung oder allgemein zum Studium zu stellen! Die Betreuer sind da, um Ihnen solche Fragen zu beantworten und Ihnen bei Bedarf zu helfen!