Heinrich Heine Universität Düsseldorf

Programmierung – WS 2025/26
Dr. Markus Brenneis
Abgabefrist für Feedback:
27.10.25, 10 Uhr

2. Übungsblatt (Lösungsvorschlag[1])

Lösungsvorschläge zu den Aufgaben von Blatt 1 finden Sie ab dem 20.10.2025 ab kurz nach 10 Uhr im Ilias. Fragen zum Lösungsvorschlag können im Ilias-Forum und im Tutorium gestellt werden. Die Übungsgruppen finden ab nächster Woche (ab 27.10.2025) statt. Denken Sie daran, dass Sie ausführlicheres Feedback zu Ihrem Code wie auf Blatt 1 beschrieben mit dem Kommentar `// Feedback gewünscht` anfordern müssen.

Aufgabe 1: Selbsttest zu Datentypen nach VL 2

Wenn der Umgang mit Datentypen noch neu für Sie ist, schauen Sie sich die Selbsttest-Aufgaben zu Datentypen und Zufallszahlen auf der Kursseite an.

Aufgabe 2: Ballistische Kurve nach VL 2

Wir wollen berechnen, in welcher Höhe sich ein Objekt nach der Zeit t befindet, wenn es von der Position x_0 aus mit einer Geschwindigkeit v_0 nach oben geworfen wird. Zum Beantworten dieser Frage haben Physiker:innen die folgende Formel hergeleitet:

$$x_0 + v_0 \cdot t - \frac{g \cdot t^2}{2}$$

Dabei ist $g = 9.81\,\mathrm{m/s^2}$ die Erdbeschleunigung.

Schreiben Sie ein Programm `Ballistic`, welches drei Variablen für die Fließkommazahlen x_0, v_0 und t definiert und das Ergebnis der Formel berechnet. Geben Sie das Ergebnis auf der Standardausgabe aus; machen Sie dabei keine weiteren Ausgaben.

Hinweis: Denken Sie daran, einen geeigneten Datentypen zum Rechnen mit Kommazahlen zu benutzen.

Für $x_0 = 2$, $v_0 = 10$ und $t = 1{,}5$ sollte Ihr Programm ungefähr 5,96 ausrechnen. Ein Körper, der in 2 Meter Höhe mit 10 m/s nach oben geworfen wird, ist also nach 1,5 Sekunden in einer Höhe von knapp 6 Metern (wenn Reibung vernachlässigt wird).

Wenn Sie Ihre Lösung vom Abgabesystem automatisch prüfen lassen wollen, müssen Sie die Vorgabe-Datei als Vorlage für Ihr Programm benutzen (Download direkt rechts neben dem Übungsblatt-Download).

[1] Bei den meisten Programmieraufgaben gibt es mehr als einen funktierenden Lösungsweg. Diskutieren Sie gerne untereinander Ihre Lösungsansätze und lerne Sie damit verschiedene Lösungsstrategien und verschiedene Anwendungsmöglichkeiten der Java-Funktionalitäten kennen. Ihre Abgabe müssen Sie aber final selbst formulieren/eintippen.

Lösungsvorschlag

- siehe `ballistic/Ballistic.java`

Aufgabe 3: Würfelsimulator nach VL 2

Korra möchte für ihre DnD-Runde ein Programm haben, mit dem sie verschiedene Arten von Würfeln simulieren kann. Dabei soll das Programm so flexibel sein, dass oben im Code in Variablen angegeben werden kann, welche Augenzahl mit dem Würfel minimal und welche maximal gewürfelt werden kann.

Schreiben Sie für Korra ein Programm `Random`, welches eine ganzzahlige Zufallszahl zwischen zwei ganzzahligen, nicht-negativen Werten erzeugt. Die erzeugte Zahl soll auch die Werte der beiden Grenzen annehmen können. Geben Sie die Zufallszahl mit abschließenden Zeilenumbruch (also mit `println`) auf der Standardausgabe aus und machen Sie keine weiteren Ausgaben.

Zur Erinnerung: Zur Erzeugung einer zufälligen Fließkommazahl zwischen 0 (einschließlich) und 1 (ausschließlich) können Sie `Math.random()` *verwenden. Jedes mal, wenn Java auf ein* `Math.random()` *stößt, wird eine neue Zufallszahl gewählt.*

Wenn ein „normaler“, sechsseitigen Würfel (W6) simuliert wird, sind untere und obere Grenze 1 bzw. 6. Das Programm soll dann zufällig eine der Zahlen 1, 2, 3, 4, 5 oder 6 ausgeben.

Anmerkungen:

- Wenn Sie Ihre Lösung vom Abgabesystem automatisch prüfen lassen wollen, müssen Sie die Vorgabe-Datei benutzen.
- Sie dürfen davon ausgehen, dass die zweite Zahl nicht kleiner als die erste ist.
- Es wäre schön, wenn alle möglichen Augenzahlen mit derselben Wahrscheinlichkeit auftreten können (Gleichverteilung); es ist aber auch in Ordnung, wenn die Zahlen mit unterschiedlichen, positiven Wahrscheinlichkeiten gewürfelt werden.

Lösungsvorschlag

- siehe `random/Random.java`

Aufgabe 4: Cosinus-Fakten nach VL 3

Zu dieser Aufgabe finden Sie ein vorgegebenes Java-Programm `Cosinus` neben dem Download dieses Übungsblatts. Das Programm funktioniert richtig und gibt Ihnen (mehr oder weniger) interessanten Fakten über die Cosinusfunktion, die Sie aus der Schule kennen, aus.

Wenn Sie in den Code schauen, merken Sie aber vielleicht, dass dieser nicht gerade übersichtlich formatiert ist. Können Sie z. B. schnell erkennen, unter welcher Bedingung die Codezeile 9 ausgeführt wird und woher das `x` in Zeile 7 kommt?

Ihre Aufgabe: Rücken Sie den vorgegebenen Code richtig ein.

Hinweise:

- Sie müssen nur Leerzeichen oder Tabulatorzeichen am Anfang der Zeilen ergänzen. Alles andere lassen Sie unverändert.
- Das korrekt eingerückte Programm muss weiterhin vollständig funktionsfähig sein.

- Sie geben die bearbeitete java-Datei als Ihre Abgabe ab.

Aufgabe 5: Schachbrett nach VL 3

Für ein text-basiertes Schachprogramm wollen wir ein einfaches Schachfeld zeichnen. Schwarze Felder werden dabei einen Stern (`*`), weiße Felder durch ein Leerzeichen repräsentiert.

Schreiben Sie ein Programm `Chess`, bei dem zu Beginn eine positive, ganze Zahl n definiert wird. Das Programm soll dann ein Schachbrett der Größe $n \times n$ auf der Standardausgabe ausgeben. Das Brett beginnt oben links mit einem schwarzen Feld.

Beispiel für $n = 5$:

```
> java Chess.java
* * *
 * *
* * *
 * *
* * *
```

Hinweise:

- Wenn Sie Ihre Abgabe mithilfe der automatischen Tests prüfen lassen wollen, benutzen Sie die Code-Vorgabe.
- Achten Sie darauf, in geraden Zeilen das letzte Leerzeichen auszugeben. Geben Sie außerdem nur genau die geforderten Leerzeichen aus.

Lösungsvorschlag

- siehe `chess/Chess.java`