Funktionale und objektorientierte Programmierkonzepte
Übungsblatt 00

TECHNISCHE UNIVERSITÄT DARMSTADT

Entwurf

Achtung: Dieses Dokument ist ein Entwurf und ist noch nicht zur Bearbeitung/Abgabe freigegeben. Es kann zu Änderungen kommen, die für die Abgabe relevant sind. Es ist möglich, dass sich **alle** Aufgaben noch grundlegend ändern. Es gibt keine Garantie, dass die Aufgaben auch in der endgültigen Version überhaupt noch vorkommen und es wird keine Rücksicht auf bereits abgegebene Lösungen genommen, die nicht die Vorgaben der endgültigen Version erfüllen.

Hausübung 00

Gesamt: 5 Punkte

Hands on mit Java & FopBot

Beachten Sie die Seite *Verbindliche Anforderungen für alle Abgaben* im Moodle-Kurs.

Verstöße gegen verbindliche Anforderungen führen zu Punktabzügen und können die korrekte Bewertung Ihrer Abgabe beeinflussen. Sofern vorhanden, müssen die in der Vorlage mit `TODO` markierten `crash`-Aufrufe entfernt werden. Andernfalls wird die jeweilige Aufgabe nicht bewertet.

Die für diese Hausübung relevanten Verzeichnisse sind `src/main/java/h00` und ggf. `src/test/java/h00`.

H1: Matrikelnummer in Moodle

?? Punkte

In dieser Aufgabe erhalten Sie einen Punkt dafür, dass Sie Ihre Matrikelnummer *fehlerfrei* in Moodle eintragen oder bereits eingetragen haben. Wenn Sie Ihre Matrikelnummer bereits in Moodle eingetragen haben, vergewissern Sie sich *dringend*, dass diese korrekt ist!

Hinweis:

In unserem Studierenden-Guide finden Sie im Abschnitt *Vorbereitung* → *Eintragen Ihrer Matrikelnummer* eine Anleitung dazu, wie Sie Ihre Matrikelnummer in Moodle eintragen.

Wenn Sie Ihre Matrikelnummer nicht oder nicht korrekt in Moodle hinterlegt haben, kann Ihre Zulassung und Ihr Bonus unter Umständen nicht korrekt verbucht werden!

H2: Einrichten von Java und Ihrer Entwicklungsumgebung

?? Punkte

Zum Bearbeiten unserer Hausübungen verwenden Sie eine Java-Entwicklungsumgebung. Hierzu zählt zum Beispiel *IntelliJ*: Die Verwendung von IntelliJ wird von uns *dringend* empfohlen und unterstützt.

Hinweis:

In unserem Studierenden-Guide finden Sie im Abschnitt

- *Vorbereitung* → *Installieren von Java* und
- *Vorbereitung* → *Installieren von IntelliJ*

Guides zur Installation von Java bzw. IntelliJ.

H3: Herunterladen und Importieren unserer Vorlagen ?? Punkte

In der FOP stellen wir Ihnen für *jede* Hausübung eine Vorlage bereit, die zur Bearbeitung der jeweiligen Hausübung verwendet werden *muss*.

> **Hinweis:**
>
> Beachten Sie weiter die auf jedem Übungsblatt am Anfang erwähnte Seite *Verbindliche Anforderungen für alle Abgaben* sowie die mitgelieferten Hinweise.

Einleitung: Geschwisterliebe

Im Zimmer, sie teilen es zu zweit,
Alfred und Kaspar, Geschwister in der Zeit.
Kaspar streut Münzen, macht das Chaos groß,
Alfred hebt sie auf, bleibt dabei nicht bloß.

Doch Kaspar spielt weiter, sein böses Spiel,
Münzen verlegt er, wild und viel.
Alfred ist müde, von der Arbeit erschöpft,
Athletisch sein, wie er es gern möcht'.

Der Zorn packt ihn, wild tanzt er nun,
Alfred lacht, hat seinen Spaß daran gewun'n.
Im Zimmer ein Tanz, voller Wut und Gier,
Geschwisterliebe versteckt sich hier.

Mit diesem Gedicht möchten wir direkt in das Thema dieses Übungsblattes einsteigen. Es beschreibt grob das Szenario, welches Sie in dieser Hausübung implementieren sollen. Dazu haben wir eine Vorlage mit der „FopBot"-Welt bereitgestellt, die Sie bitte für Ihre Abgabe verwenden. Keine Sorge, eine Gedichtsanalyse steht hier nicht auf der Agenda.

Die zwei Roboter Alfred und Kaspar sind Geschwister, die unterschiedlicher nicht sein könnten. Während Kaspar gerne Chaos stiftet und die Welt mit seinen Münzen verschmutzt, ist Alfred hingegen ein disziplinierter Ordnungshüter. Alfred setzt sich dafür ein, die Verschmutzung der „FopBot"-Welt durch Kaspar zu verhindern. Immer wenn Kaspar Unruhe stiftet, fühlt sich Alfred gezwungen, dem entgegen zu wirken. Sprich, er will und soll die Münzen aufsammeln.

Allerdings ist er ein Tagträumer und schaut gerne aus der Welt hinaus. Dabei bemerkt er allerdings nicht, wenn sein Bruder Kaspar wieder Unordnung stiftet. Deshalb hat er sich eine Strategie überlegt, wie er die Münzen am effizientesten aufsammeln kann, wenn sein Bruder wieder Unordnung stiftet.

Wir wollen nun versuchen, dieses Szenario nachzuahmen…

> **Wir verfolgen „Abschreiben" und andere Arten von Täuschungsversuchen!**
>
> Disziplinarische Maßnahmen treffen nicht nur die, die abschreiben, sondern auch die, die abschreiben lassen! Beachten Sie die Seite *Grundregeln der Wissenschaftsethik* des Fachbereichs Informatik.

H4: Kaspar und Alfred ?? Punkte

Aus der Vorlesung kennen Sie die „FopBot“-Welt. In dieser Hausübung implementieren Sie ein erstes Java-Programm, das zwei Roboter erzeugt, durch die Welt bewegt und dabei Münzen ablegt.

In der Vorlage zu dieser Hausübung haben wir bereits zwei Anweisungen vorgegeben, die einen Roboter `kaspar` und einen Roboter `alfred` einrichten, wobei `kaspar` in der unteren linken Ecke der Welt mit Blickrichtung nach links startet und `alfred` in der oberen rechten Ecke der Welt mit Blickrichtung nach rechts startet:

```
1 Robot kaspar = new Robot(0, 0, LEFT, 20);
2 Robot alfred = new Robot(4, 4, RIGHT, 0);
```

Hinweis:

Screenshots aus der Welt der Roboter dürfen Sie unbedenklich mit anderen Studierenden teilen – nur eben nicht den Quelltext oder eine übersetzte Variante des Quelltexts!

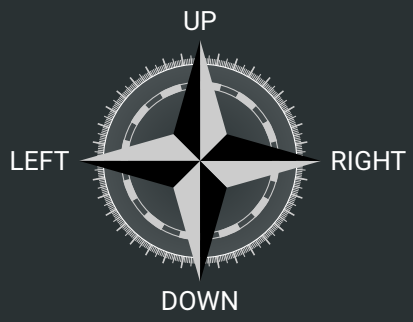
Hinweis:

Sie dürfen in dieser Hausübung davon ausgehen, dass die Welt immer die Größe 5x5 besitzt.

Erinnerung:

Bei „FOPBot“ wird die Position des Roboters in der Welt durch eine X- und eine Y-Koordinate dargestellt. Die Koordinaten beginnen bei 0 und die X-Koordinaten starten von links, die Y-Koordinaten starten von unten.

(a) Koordinaten einer „FOPBot“-Welt



(b) Richtungen in der FOPBot-Welt

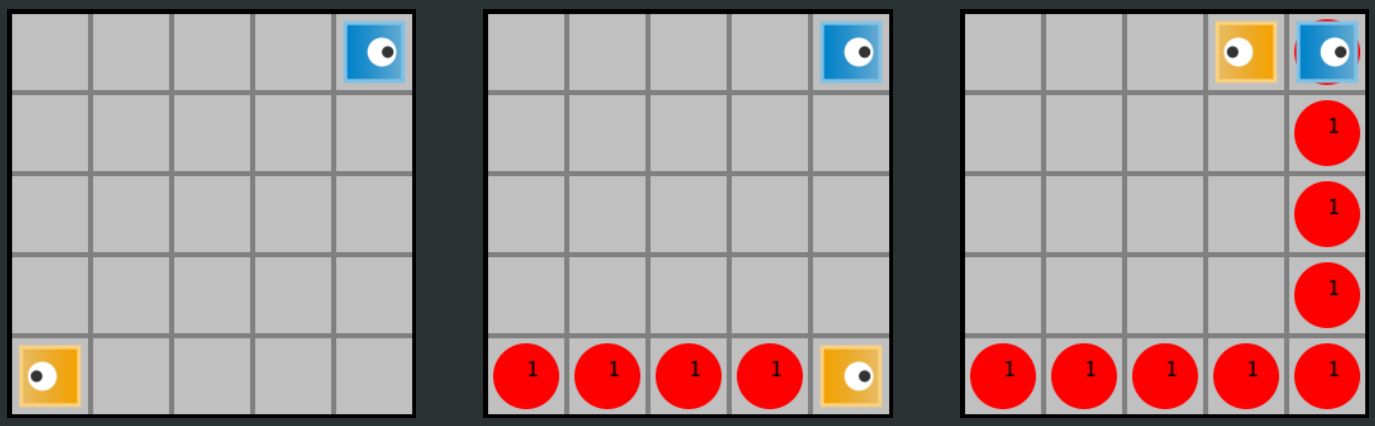
Abbildung 1: Erinnerung zur FOPBot-Welt

H4.1: Kaspar, der Zimmerchaot ?? Punkte

Als erstes ist Kaspar an der Reihe. Er startet in der unteren linken Ecke $(0, 0)$ und soll seine Münzen so in der Welt ablegen, dass das endgültige Muster einem spiegelverkehrten „L“ ähnelt. Dafür soll er sich zuerst so oft nach rechts drehen, bis er nach rechts schaut. Danach soll er sich entlang der unteren Wand so lange vorwärts bewegen, bis er auf eine Wand stößt, und dabei **vor** jedem Vorwärtschritt **eine** Münze ablegen. An der unteren rechten Ecke der Welt angekommen, soll er sich nun einmal nach links drehen und anschließend seinen Weg zur oberen rechten Ecke der Welt fortsetzen. Dabei soll er wieder **vor** jedem Schritt eine Münze ablegen.

An der oberen rechten Ecke, wo auch Alfred platziert ist, soll Kaspar sich abschließend einmal nach links drehen, eine Münze ablegen und einen Vorwärtschritt machen (wir wollen den armen Alfred nicht „verdecken“), wodurch am Ende auch das spiegelverkehrte „L“-Münzenmuster vervollständigt wird. Anhand der Abbildungen 2 können Sie entnehmen, wie es beabsichtigt ist.

Implementieren Sie nun anhand der obigen Beschreibung die Bewegung von Kaspar in Java-Code. Beachten Sie, dass in der FopBot-Welt zwei Roboter mit unterschiedlicher Position und Blickrichtung positioniert wurden. Anhand der Bezeichner der Roboter sollte Ihnen klar sein, welcher Roboter-Instanz Sie ansprechen und welche Methoden Sie anwenden müssen.



(a) Anfangszustand:
- Roboter Kaspar(0,0); Blickrichtung **links**
- Roboter Alfred(4,4); Blickrichtung **rechts**

(b) Zwischenzustand (Kaspar erreicht die rechte unteren Ecke):
- Roboter Kaspar(4,0);
- Roboter Alfred(4,4);

(c) Endzustand:
- Roboter Kaspar(3,4); Blickrichtung **links**
- Roboter Alfred(4,4)

Abbildung 2: Bewegung von Kaspar

Verbindliche Anforderungen:

Nutzen Sie für die Bewegung entlang der Wände genau zwei `for`-Schleifen. Die Schleifen werden verwendet, um die Bewegung entlang der Wände zu implementieren.

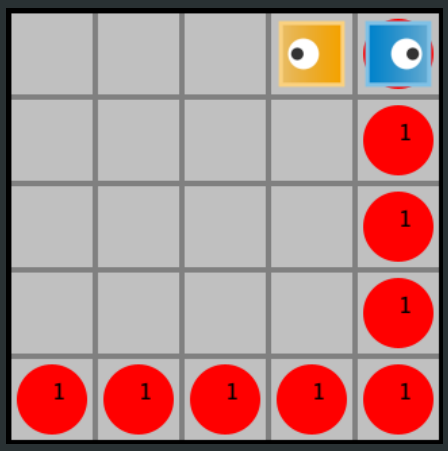
H4.2: Alfred, der Ordnungsfanatiker ?? Punkte

Nun ist **Alfred** an der Reihe, sich in Bewegung zu setzen und das Chaos, welches sein Bruder in der „FopBot“-Welt verursacht hat, zu beseitigen. Dafür soll er sich zuerst so oft nach unten drehen, bis er nach unten schaut. Nachdem er die richtige Blickrichtung hat, soll er sich entlang der rechten Wand so lange vorwärts bewegen, bis er eine Wand vor sich hat und dabei **vor** jedem Vorwärtschritt **eine** Münze **aufheben**. Danach soll er sich solange drehen, bis er nach links schaut, und die vorherige Bewegung wiederholen.

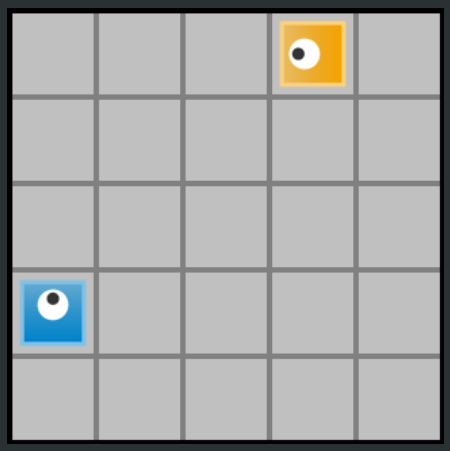
Nun befindet sich Alfred in der linken unteren Ecke der Welt. Zum Abschluss soll er sich oft drehen, bis er nach oben schaut, eine Münze aufheben und seinen letzten Schritt nach vorne machen, wodurch am Ende auch das „L“-Münzenmuster beseitigt sein sollte.

Zum großen Abschluss soll Kaspar, so frech wie er ist, seine restlichen Münzen ablegen, um Alfred (wieder) aus der Fassung zu bringen. Hierbei „tanzt“ bzw. dreht sich Alfred **solange** nach links, bis Kaspar keine Münzen mehr hat. Anders formuliert: Nach jeder Münzplatzierung durch Kaspar soll Alfred genau eine Linksdrehung durchführen, solange Kaspar noch Münzen hat. Kaspar bewegt sich dabei gar nicht. (Münzablage zählt nicht als Bewegung)

Implementieren Sie nun anhand der obigen Beschreibung die Bewegungen von Alfred und Kaspar in Java-Code.



(a) Anfangszustand:
- Roboter Kaspar(3,4); Blickrichtung **links**
- Roboter Alfred(4,4); Blickrichtung **rechts**



(b) Endzustand:
- Roboter Kaspar(3,4)
- Roboter Alfred(1,0)

Abbildung 3: Bewegung von Alfred

Verbindliche Anforderungen:

Die Nutzung von `for`-Schleifen ist hier untersagt. Verwenden Sie stattdessen genau sechs `while`-Schleifen.

Hinweis:

Folgende Methoden aus der FopBot-Framework sei Ihnen für diese Aufgabe vorgestellt:

- `<roboterInstanz>.isFrontClear()` gibt den boolschen Wert `true` zurück, wenn der Roboter vor ihm keine Wand vor sich hat.
- `<roboterInstanz>.isFacingDown()` gibt den boolschen Wert `true` zurück, wenn der Roboter nach unten schaut (vs. `isFacingLeft` vs. `isFacingRight` vs. `isFacingUp`).
- `<roboterInstanz>.hasAnyCoins()` gibt den boolschen Wert `true` zurück, wenn der Roboter noch Münzen (zum Ablegen) besitzt.

(„<roboterInstanz>“ sei hier als ein Platzhalter für den jeweiligen Roboter gedacht)