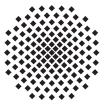

Universität Stuttgart

RSS
Reliable Software Systems

Übungsblatt 9

Programmierung und Softwareentwicklung (WiSe 2019/2020)

Abgabe: Fr. 20.12.2019, 23:59 Uhr — Besprechung: KW 3

- Bitte lösen Sie die Übungsaufgabe in **Gruppen von 2 Studenten**.
- Dieses Übungsblatt besteht aus zwei Teilen (A+B). Teil A ist in der Präsenzübung zu lösen. Teil B ist in Heimarbeit (Gruppe von 2 Studenten) zu lösen und rechtzeitig abzugeben.
- Geben Sie zu Beginn der Dateien Ihre Namen (Vor- und Nachname), die Matrikelnummern und die E-Mail-Adressen an.
- Benennen Sie die Dateien nach dem folgenden Schema:
  1. **PSE[ÜB-Nr]-[Nachnamen der Teammitglieder]-[Nachname des Tutors].pdf**
  2. **PSE[ÜB-Nr]-[Nachnamen der Teammitglieder]-[Nachname des Tutors].zip**
- Das zip soll Ihr Eclipse Projekt mit allen Änderungen enthalten. Sie können das zip in Eclipse wie folgt erstellen: File - Export - General - Archive File - ...

Lernziel: Dieses Übungsblatt dient dazu, dass die in der Vorlesung vorgestellten Konzepte der Objektorientierung und Vererbung vertieft und angewendet werden. Zudem kennen Sie bereits ein umfangreiches Set von Stilrichtlinien, welche Sie anwenden, üben und vertiefen sollen.

Vorbereitung: Stellen Sie sicher, dass Sie sich mit den Vorlesungsunterlagen zum Thema Vererbung vertraut gemacht haben und die Übungsblätter 7 und 8 absolviert haben.

Punkte: Dieses Übungsblatt enthält zwei Teile. Teil A mit 3 Aufgaben und Teil B mit 2 Aufgaben. Im Teil B können Sie bis zu 56 Punkte erzielen.

Style: Bitte halten Sie die in der Vorlesung vorgestellten Style-Regeln ein. Dazu gehören unter anderem JavaDoc, Vor- und Nachbedingungen, Prüfung derer mittels offensiver (mittels Asserts) und defensiver Checks (mittels Exceptions), Schleifeninvarianten und -varianten (mittels natürlicher Sprache oder JML im Kommentarblock über der Schleife), Sichtbarkeiten oder Read-Only Restriktionen. Der Style Ihrer Implementierung wird mit bis zu 50% bewertet.

Unterlagen: Alle relevanten Unterlagen für die Bearbeitung des Übungsblatts finden Sie wie gehabt in unserem git-Repository: `https://github.com/RSS-PSE-WS1920/`
Die Dokumentation der Hamster-API finden Sie wie gewohnt hier: `http://caloundra.informatik.uni-stuttgart.de` (nur aus dem Uni-Netz erreichbar).

Viel Erfolg!

1 Teil A - Präsenzaufgaben

In dieser Übung vertiefen wir die Kozepte der Vererbung und Polymorphie. Dazu sollten Ihnen mindestens die Vorlesungsunterlagen zu dem Thema vertraut sein.

Aufgabe 1 Vererbung II und Polymorphie

In dieser Übung sollen Sie die Klasse `Hamster` "richtig" erweitern. Öffnen Sie dazu das Projekt zu Übungsbatt 9 aus dem Repository.

(a) Legen Sie ein neues Package mit dem Namen `de.unistuttgart.iste.rss.oo.hamstersimulator.exercise9.presence.polymorphy` an.

(b) Erstellen Sie in dem Package eine neue Klasse `MyGreatHamster`.

(c) Erweitern Sie die Klasse so, dass Sie von der Klasse `Hamster` erbt.

(d) Fügen Sie mindestens drei Operationen hinzu, die wir in den letzten Übungen kennengelernt haben (z.B. pickAll, turnRight, etc.). `Hinweis:` Die Operationen sollen dabei so implementiert werden, dass sie für beliebige der spezialisierten Hamsterobjekte funktionieren.

Aufgabe 2 Racing Hamster

In dieser Aufgabe wollen wir Polymorphie richtig nutzen. Betrachten Sie dazu die abstrakte Klasse `RacingHamster` und die Klasse `HamsterRaceGame`.

(a) Schauen Sie sich zunächst die Implementierung der Klasse `RacingHamster` an und versuchen Sie, diese zu verstehen.

(b) Implementieren Sie nun zwei Rennhamster (`NormalRacingHamster` und `FastRacingHamster`), die von der abstrakten Klasse erben und die entsprechenden Operationen implementieren.

(c) Öffnen Sie die Klasse `HamsterRaceGame` und implementieren Sie die Operation `initRace`, in welcher ein schneller und ein normaler Rennhamster initialisiert werden soll.

(d) Führen Sie als nächstes die Operation `main` der Klasse `Main` aus. Was beobachten Sie? Verhält sich das Programm korrekt? Begründen Sie Ihre Antwort.

(e) Diskutieren Sie innerhalb Ihrer Gruppe, wo in Ihrem Programm Polymorphie und wo dynamsiches Binden genutzt wird.

(f) Denken Sie wie immer an Vor- und Nachbedingungen sowie auch an JavaDoc für Ihre eigenen Operationen, sowie für die bereitgestellten Operationen.

(g) **Herausforderung:** Passen Sie Ihre oder die gegebenen Klassen an, damit sich ihr Programm korrekt verhält und passen Sie ggf. Vor- und Nachbedingen an.

Aufgabe 3 FIFO-Datenstruktur

Diese Übungsaufgabe befasst sich mit Datenstrukturen. Im konkreten Fall soll sich Ihr Hamster (Paule) in einem ihm unbekannten Territorium befinden. Unabhänig von der Blickrichtung soll er sich solange vorwärts bewegen, bis er auf eine Wand trifft. Auf dem Weg soll er alle Körner aufsammeln und sich dabei merken, wieviele Körner er auf jedem Feld eingesammelt hat.

Zusätzlich zu dem Merken der Körner soll sich Paule um 180 Grad drehen, sobald er auf eine Mauer trifft. Anschließend soll er die Körner in genau der gleichen Reihenfolge ablegen, in der er sie aufgesammelt hat (first in first out). Als Ergebnis liegen die Körner am Ende in genau umgekehrter Reihenfolge auf dem Feld und Paule steht wieder auf seinem Ursprungsfeld.

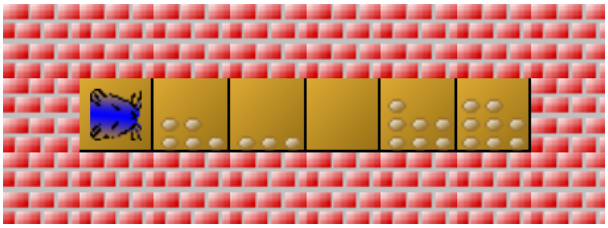

Abbildung 1: Execution of the hamster task

(a) Überlegen Sie sich mit Ihrem Teampartner, welche Datenstruktur dafür geeignet ist.

(b) Überlegen Sie sich welche Hilfsoperationen Sie für diese Aufgabe brauchen und beachten Sie auch die Ihnen bereits bekannten Hilfsklassen aus den vergangenen Übungen.

(c) Öffnen Sie die Klasse `MemoryHamster` und implementieren Sie die Operation `reverseOrder`.

(d) Achten Sie auf die Style-Richtlinien und definieren Sie Vor- und Nachbedingungen.

Aufgabe 4 LIFO-Datenstruktur

Im Folgenden wollen wir, dass Paule die Körner genauso wieder auf das Feld legt, wie er sie vorgefunden hat (last in first out).

(a) Überlegen Sie sich mit Ihrem Teampartner, welche Datenstruktur dafür geeignet ist.

(b) Überlegen Sie sich, welche Hilfsoperationen Sie für diese Aufgabe brauchen und beachten Sie auch die Ihnen bereits bekannten Hilfsklassen aus den vergangenen Übungen.

(c) Öffnen Sie die Klasse `MemoryHamster` und implementieren Sie die Operation `inOrder`.

(d) Achten Sie auf die Style-Richtlinien und definieren Sie Vor- und Nachbedingungen.

2 Teil B - Heimarbeit

Aufgabe 1 Mutable und Immutable

Falls noch nicht geschehen, laden Sie sich das Git-Repository für dieses Übungsblatt herunter.

In der folgenden Aufgabe sollen Sie nochmal die Konzepte von unveränderlichen Klassen wiederholen. Dazu sollen Sie zuerst eine veränderliche Klasse `EMail` schreiben. Im Anschluss eine Klasse `EMailArchive`, der nur E-Mails hinzugefügt werden können.

(a) *(6 Punkte)* Erstellen Sie die Klasse `EMail` im Paket `immutable`. Eine E-Mail hat einen Sender, Empfänger, Betreff und einen Nachrichtenteil. Erstellen Sie diese Klasse und alle zugehörigen Getter- und Setter-Operationen.

(b) *(8 Punkte)* Erstellen Sie eine Klasse `EMailArchive` im Paket `immutable`. Jedes Archiv hat ein Erstellungsdatum, das automatisch beim Erzeugen gesetzt wird, sowie einen Namen und eine sortierte Liste (also eine Datenstruktur, die die Reihenfolge der enthaltenen Objekte beachtet), die Objekte vom Typ `EMail` enthält. Name und Liste sollen dem Konstruktor übergeben werden. Machen Sie die Klasse unveränderlich und geben Sie ihr Getter-Operationen. Beachten Sie, dass es momentan noch nicht möglich sein soll, E-Mails hinzuzufügen, nachdem der Konstruktor ausgeführt wurde, oder bereits der Liste hinzugefügte E-Mails zu verändern oder zu löschen.

(c) *(6 Punkte)* Fügen Sie Ihrer Klasse eine Operation `addEMailToArchive(final EMail mail)` hinzu, die es ermöglicht, eine E-Mail der Liste hinzuzufügen. Weiterhin soll aber gelten, dass bereits hinzugefügte Mails nicht mehr verändert oder gelöscht werden können.

Aufgabe 2 Polymorphie

Öffnen Sie die Klasse `MyCheaterHamster`. Diese Klasse überschreibt das Kommando `move`. Schauen Sie sich die Implementierung an. In Listing 1 finden Sie die Implementierung des Befehls `move` in der Klasse `Hamster` sowie die entsprechenden Vor- und Nachbedingungen.

(a) *(2 Punkte)* Geben Sie an, was die Operation `move` im `MyCheaterHamster` machen soll.

(b) *(2 Punkte)* Geben Sie an, ob die Operation `move` im `MyCheaterHamster` funktioniert und ob das Ziel damit erreicht wird. Begründen Sie ihre Antwort kurz.

(c) *(2 Punkte)* Begründen Sie, warum diese Art der Implementierung dem Konzept der objektorientierten Programmierung widerspricht.

```
/*@
 @ requires frontIsClear();
 @ requires isInitialized;
 @ ensures getDirection() == Direction.NORTH ==>
 @       \old(getLocation()).getRow() == getLocation().getRow() + 1
    &&
 @       \old(getLocation()).getColumn() == getLocation().getColumn()
    ;
 @ ensures getDirection() == Direction.SOUTH ==>
 @       \old(getLocation()).getRow() == getLocation().getRow() - 1
    &&
 @       \old(getLocation()).getColumn() == getLocation().getColumn()
    ;
 @ ensures getDirection() == Direction.EAST ==>
 @       \old(getLocation()).getRow() == getLocation().getRow() &&
 @       \old(getLocation()).getColumn() == getLocation().getColumn()
     - 1;
 @ ensures getDirection() == Direction.WEST ==>
 @       \old(getLocation()).getRow() == getLocation().getRow() &&
 @       \old(getLocation()).getColumn() == getLocation().getColumn()
     + 1;
 @*/
/**
 * Move the hamster one step towards its looking direction.
 * @throws FrontBlockedException When the tile in front of the
```

```
     * hamster is blocked
     */
    public void move() {
        this.game.processCommandSpecification(
                new MoveCommandSpecification(this.internalHamster));
    }
```

Listing 1: Implementierung des Befehls `move` in `Hamster`

Aufgabe 3 Bibliothek von Alexandria

Neben der Universitätsbibliothek besitzt jeder Lehrstuhl zusätzlich einen Handapparat (eine kleine eigene Bibliothek). Um eine bessere Übersicht zu erhalten, welche Medien sich in unserem Handapparat befinden, wer aktuell welches Medium ausgeliehen hat und wann es zurückgegeben werden muss, wollen wir ein Bibliotheksmanagementsystem entwickeln. Da wir begeistert von dem objektorientierten Ansatz sind, sollen Sie für uns im Folgenden den Entwurf erstellen.

Dazu erhalten Sie folgende **Szenariobeschreibung**:

In unserem Handapparat befinden sich unterschiedliche Medien: Printmedien und Multimedia-Dateien. Bei Printmedien haben wir Paper, Journals, Zeitungen und Bücher; Multimedia-Dateien lassen sich in Videos und Podcasts unterscheiden.

Zusätzlich gibt es unterschiedliche Personen, die Zugriff auf den Handapparat haben: Studenten, Mitarbeiter und Externe. Mitarbeiter dürfen Medien für unbestimmte Zeit ausleihen, während Studenten eine maximale Ausleihdauer von 3 Monaten haben und Externe von 30 Tagen.

Zusätzlich haben wir aktuell ein Buch, in dem steht, welche Person welches Medium ausgeliehen hat, ob es bereits zurückgebracht wurde und bis wann es spätestens zurückgebracht werden muss.

(a) *(10 Punkte)* Analysieren Sie die Szenariobeschreibung und identifizieren Sie alle benötigten Klassen, deren Attribute sowie Abfragen und Kommandos. Erstellen Sie weiter eine Vererbungshierachie und identifizieren Sie mindestens zwei abstrakte Klassen.

(b) *(20 Punkte)* Setzen Sie Ihren Entwurf in Eclipse um und achten Sie dabei auf alle Styleregeln (Sichtbarkeiten, Vor- Nachbedingungen, JavaDoc, defensives und offensives Programmieren – wo benötigt).

Hinweis: Sie sollen nicht alle Operationen mit Logik ausimplementieren, sondern lediglich die im Entwurf identifizierten Bestandteile angeben.

3 Teil L - Lehramtsteil

Dieser Teil enthält eine Präsenzaufgabe die speziell für Lehramtsstudierende konzipiert ist. Das bedeutet allerdings nicht, dass "Nicht-Lehramtsstudierende" diese nicht auch bearbeiten dürfen. Sie bezieht sich auf den Vorlesungsstoff und die Übungsaufgaben, betrachtet diese aber aus einer didaktischen Perspektive.

Aufgabe 1 Vererbung

Vererbung ist ein wichtiges Konzept in der objektorientierten Programmierung. Allerdings ist sie auch ein sehr abstraktes und theoretisches Konzept. Diese Aufgabe soll Sie Ideen entwickeln lassen, wie man Vererbung in der Schule einführen und nutzen kann.

Hinweis: Auch in dieser Aufgabe wird das Prinzip der "didaktischen Reduktion" angewendet.

(a) Überlegen Sie sich mit Ihrem Team, welche Konzepte der Vererbung Sie in der Vorlesung kennen gelernt haben (Sie sollten auf mindestens vier Konzepte kommen).

(b) Diskutieren Sie mit Ihrem Team, welche dieser Konzepte für die Schule geeignet und welche ungeeignet sind. Begründen Sie Ihre Entscheidung.

(c) Entwickeln Sie mit Ihrem Team Unterrichtsideen für die *geeigneten* Vererbungskonzepte. Denken Sie hier an mögliche (kleine) Aufgaben oder Beispiele zur Erklärung.

Hinweise zu den Übungen:

- Die Abgabe erfolgt im ILIAS.
- Durch die Teilnahme am Übungsbetrieb können Sie sich für die Teilnahme an der Klausur qualifizieren.
  - Bestehen von min. 80% aller Übungsblätter.
  - Ein Übungsblatt gilt als bestanden, wenn 50% der Punkte erreicht wurden.
  - Teilnahme an min. 80% der Übungen.
  - Bestehen der Scheinklausur.

Viel Erfolg!