Σ 14 Punkte

Programmierung für Alle — Abgabe 10

Statische Member und Exceptions

Abgabefrist: Montag, 06. Januar 2020, 10 Uhr

Beschreibung

In der Woche vor der Präsenzübung sind statische Klassenmember und Exceptions am bekannten Flugzeug-Beispiel vorgestellt worden. In diesem Übungsblatt werden Sie das Flugzeug-Programm nach den vorgestellten Prinzipien umbauen.

Aufgaben

1. Flugzeug Abstürze verhindern (14 Punkte)

Bisher konnten unsere Flugzeuge nur starten und landen. Jetzt, wo Fliegen auch eine Methode geworden ist, wird es immer schwieriger unfallfrei zu bleiben. Für alle folgenden Aufgaben gilt, dass jedes Flugzeug 10 mal starten, fliegen und landen muss ohne einen Absturz zu verursachen. Es darf aber sein, dass nach dem Starten, während des Fliegens, Probleme auftreten und behoben werden. In diesem Fall wurde natürlich weder geflogen noch gelandet.

a) Wir benötigen eine Werkstatt! (3 Punkte)

Erstellen Sie die Datei Werkstatt.java. In dieser Datei soll eine Klasse Werkstatt geschrieben werden. Unsere Werkstatt soll zwei Methoden `flugzeugReparieren` und `flugzeugTanken` haben.

Die Methode `flugzeugReparieren` muss ein Flugzeug-Objekt als Parameter annehmen und auf diesem Flugzeug die `reparieren()`-Methode aufrufen. Danach soll "Das Flugzeug wurde repariert." ausgegeben werden.

Die Methode `flugzeugTanken` muss ein Objekt, das getankt werden kann, als Parameter annehmen und auf diesem die `tanken()`-Methode aufrufen. Danach soll "Es wurde getankt." ausgegeben werden.

Beide Methoden ändern nichts an der Werkstatt und hängen auch nicht von einem bestimmten Werkstatt-Objekt ab, deshalb sollen Sie statisch sein.

b) Leider kaputt! (5 Punkte)

Wenn Sie das Programm im aktuellen Zustand compilieren und ausführen stürzt direkt das Modellflugzeug ab. Der Grund ist, dass **alle** Flugzeuge zu Beginn kaputt sind. Im Code sind vier Stellen markiert, an denen entweder Änderungen durchgeführt oder neuer Code hinzugefügt werden kann.

Erstellen Sie eine Datei LeiderKaputt.java, in der sie eine Exception-Klasse mit gleichem Namen schreiben. Nutzen Sie diese Excpetion an den richtigen Stellen in den `fliegen()`-Methoden der verschiedenen Flugzeuge. Die Exception soll geworfen werden wenn das Flugzeug kaputt ist.

Passen Sie zusätzlich die Klasse `FlugzeugTest` an. An der Stelle, die die `fliegen()`-Methode der einzelnen Flugzeuge aufruft, müssen Sie ihre `LeiderKaputt`-Exception fangen und behandeln, indem Sie das Flugzeug von der Werkstatt reparieren lassen. Achten Sie darauf, dass Flugzeuge nur repariert werden, wenn genau diese Exception aufgetreten ist. Es ist also keine Lösung der Aufgabe wenn Sie einfach alle Exceptions gleich behandeln.

c) Kein Benzin mehr! (6 Punkte)

Wenn Sie das Programm jetzt compilieren und ausführen, stürzt direkt die B787 ab. Der Grund ist, dass **alle tankbaren** Flugzeuge zu Beginn nicht betankt sind. Im Code sind vier Stellen markiert, an denen entweder Änderungen durchgeführt oder neuer Code hinzugefügt werden kann.

Erstellen Sie eine Datei KeinBenzinMehr.java, in der sie eine Exception-Klasse mit gleichem Namen schreiben. Nutzen Sie diese Excpetion an den richtigen Stellen in den `fliegen()`-Methoden der verschiedenen Flugzeuge. Die Exception soll geworfen werden wenn der Tank der Flugzeuge leer ist.

Passen Sie zusätzlich die Klasse `FlugzeugTest` an. An der Stelle, die die `fliegen()`-Methode der einzelnen Flugzeuge aufruft, müssen Sie ihre `KeinBenzinMehr`-Exception fangen und behandeln, indem Sie das Flugzeug von der Werkstatt betanken lassen. Achten Sie darauf, dass Flugzeuge nur betankt werden, wenn genau diese Exception aufgetreten ist. Es ist also keine Lösung der Aufgabe wenn Sie einfach alle Exceptions gleich behandeln. Zusätzlich muss auch weiterhin die `LeiderKaputt`-Exception funktionieren wie in b) angegeben.

Tipp: Eventuell müssen Sie hier Objekte casten. Auch Interfaces sind gültige Typen für Variablen und Casting.

Abgabeformat

Packen Sie alle Dateien in ein **ZIP-Archiv** namens **Gruppe_XXX.zip.** Ersetzen Sie dabei XXX durch Ihre **Gruppennummer.**

Das ZIP-Archiv enthält:

- **zwölf** Dateien **FlugzeugTest.java, Flugzeug.java, ModellFlugzeug.java, VerkehrsFlugzeug.java, Boeing.java, B787.java, Airbus.java, A380.java, KannTanken.java, Werkstatt.java, LeiderKaputt.java und KeinBenzinMehr.java** mit Ihrem kommentierten Code aus Aufgabe 1.

Wichtig: Bevor Sie Ihren Code hochladen, empfehlen wir einmal javac *.java im Ordner auszuführen, damit auch wirklich alle Dateien einmal in ihrer letzten Version kompiliert werden. Es ist in den vorherigen Abgaben immer wieder aufgetreten, dass wir Code nicht kompilieren konnten, weil aus Versehen doch noch Änderungen in Dateien gemacht worden waren, die dann nicht mehr kompilierten.

Bewertung

Bearbeiten Sie die Aufgaben sorgfältig, denn wir bewerten auch Ihren gewollten Versuch in der Punkte-Vergabe positiv.

Um das Übungsblatt zu bestehen, müssen Sie mindestens 50% der Punkte erreichen.

Code, den wir nicht kompilieren können, bewerten wir mit 0 Punkten.

Für Abgaben, die nicht dem Abgabeformat entsprechen, ziehen wir 1 Punkt ab. Dazu müssen Sie auch wieder Ihre **.class-Dateien** entfernen.

Für Codedateien ohne Kommentare ziehen wir 1 Punkt ab.

Abgaben nach der Abgabefrist sind nicht bestanden. Dies gilt auch für Plagiate, die zusätzlich zu weiteren Konsequenzen führen können.