Übung 3 zur OOS WS19 Besprechung am 30.10.2019

Aufgabe 41 wird in Form eines Tutoriums durchgeführt.

Der Besuch des Tutoriums ist von enormer Bedeutung, da es sich bei dieser Aufgabe um den Aufgabentyp API-Entwicklung handelt, der in Etwa die Hälfte der Punktzahl in einer Klausur ausmacht.

Aufgrund dessen wird die Lösung der Aufgabe zusammen im Tutorium erarbeitet.

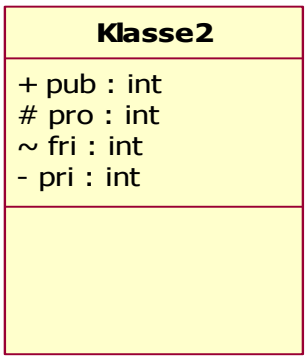
Aufgabe 14: (Pakete und Sichtbarkeit)

Führen Sie einige Experimente zur Sichtbarkeit von Attributen durch.

a) Erzeugen Sie ein Paket mit Namen `übung3`.

b) Erzeugen Sie in diesem Paket eine `public`-Klasse `Klasse1` mit `main`-Methode.

c) Implementieren Sie in der Datei `Klasse1.java` eine Klasse `Klasse2` gemäß der UML-Beschreibung:



d) Erzeugen Sie in der `main`-Methode von `Klasse1` ein Objekt von `Klasse2` mit Namen `klasse2`.

e) Initialisieren Sie die 4 Attribute von `klasse2` mit 0, falls möglich und begründen Sie für welche der 4 Attribute das nicht möglich ist.

f) Erzeugen Sie im gleichen Paket, jedoch in der Datei `Klasse3.java` eine Klasse `Klasse3` ohne Modifier `public` gemäß der UML-Beschreibung von `Klasse2` in Teil c).

h) Erzeugen Sie in der `main`-Methode von `Klasse1` ein Objekt von `Klasse3` mit Namen `klasse3`. Begründen Sie, ob dies möglich ist.

i) Initialisieren Sie die 4 Attribute von `klasse3` mit 0, falls möglich und begründen Sie für welche der 4 Attribute das nicht möglich ist.

j) Was passiert, wenn Sie in der Datei `Klasse1.java` den `package`-Befehl löschen?

k) Was passiert, wenn Sie auch in der Datei `Klasse3.java` den `package`-Befehl löschen?

l) Erzeugen Sie ein weiteres Paket mit Namen `übung3a`.

m) Erzeugen Sie in diesem Paket in der Datei `Klasse4.java` eine Klasse `Klasse4` ohne Modifier `public` gemäß der UML-Beschreibung von `Klasse2` in Teil c).

n) Erzeugen Sie in der `main`-Methode von `Klasse1` ein Objekt von `Klasse4` mit Namen `klasse4`. Begründen Sie, ob dies möglich ist.

Übung 3 zur OOS WS19 Besprechung am 30.10.2019

Aufgabe 15: (Sichtbarkeit)

Definieren Sie eine Klasse `NurSelbst`, die als einziges in der Lage ist, Objekte vom eigenen Typ zu erzeugen und überprüfen Sie, dass wirklich keine andere Klasse in der Lage ist, solche Objekte zu erzeugen.

Aufgabe 17: (Klassenattribute und -methoden)

Erweitern Sie die Klasse `Auto` um eine Klassenmethode `kmDurchschnitt`, die die durchschnittliche km-Leistung aller von der Klasse `Auto` erzeugten Instanzen liefert.

Verwenden Sie hierzu:

- ein Klassenattribut `anzErzeugteAutos`, das die Anzahl aller von der Klasse `Auto` erzeugten Instanzen liefert und
- ein Klassenattribut `gesamtLeistung`, das die Gesamt-km-Leistung aller von der Klasse `Auto` erzeugten Instanzen liefert.

Zur korrekten Berechnung des Klassenattributs `gesamtLeistung` erlauben Sie die Änderung des Attributs `kmStand` der Klasse Auto nur über eine Methode `setkmStand(intkmStand)`, die immer dann, wenn sich der `kmStand` einer Instanz ändert, auch den Wert von `gesamtLeistung` ändert. Die Methode `kmDurchschnitt` liefert dann den Quotient aus den beiden Klassenattributen.

Testen Sie die Klassenmethode `kmDurchschnitt`, indem Sie in einem main-Programm 5 Autos erzeugen, deren kmStand mindestens einmal verändert wird, und am Ende `kmDurchschnitt` aufrufen.

Aufgabe 41: (lineare Liste als abstrakter Datentyp) wird im Tutorium besprochen

Eine *lineare Liste* ist eine potenziell unendliche Datenstruktur zur Speicherung von Objekten. Sie implementiert den Zugriff auf ihre Elemente, indem man beim ersten Element beginnt und dann linear jeweils zum nächsten läuft.

Ein *abstrakter Datentyp* ist eine Klasse, der seine eigentliche Implementierung verbirgt (kapselt) und den Zugriff nur über Zugriffsmethoden zulässt.

Implementieren Sie eine lineare Liste für beliebige Objekte als Einträge in Java als abstrakten Datentyp, indem Sie eine minimale Anzahl von Methoden zur Verfügung stellen, damit der Anwender auf der linearen Liste operieren kann. Minimal bedeutet in diesem Zusammenhang, dass sich eine enthaltene Methode nicht ausschließlich durch andere implementieren lässt.

Hinweis: Gehen Sie von der vordefinierten Listenstruktur in Prolog aus und überlegen Sie sich, wie Sie diese in Java realisieren können.

Überlegen Sie sich hierzu auch geeignete Exceptions, die Sie selbst implementieren und an den richtigen Stellen werfen.

Ergänzen Sie Ihre Implementierung um weitere sinnvolle Methoden, die über die minimale Implementierung hinausgehen.

Übung 3 zur OOS WS19 Besprechung am 30.10.2019

Überlegen Sie sich sinnvolle Tests und führen Sie diese in einem `main`-Programm aus.

Programmieren Sie so, dass Sie immer dann, wenn Sie eine Methode benötigen, die noch nicht da ist, zuerst den Aufruf an der Stelle, wo sie benötigt wird, eintragen und sich dann von Eclipse die Methode erzeugen lassen und erst dann implementieren.

Hierzu können Sie auch schon die Tests vor den Methoden implementieren.