Abgabe-Übungsblatt A2

Die Lösungen für dieses Übungsblatt sollen abgegeben werden. Die erzielten Punkte werden für die Studienleistung angerechnet. Durch dieses Übungsblatt können bis 8 Punkte erworben werden. Insgesamt können bis zu 40 Punkte erworben werden. Die Studienleistung ist bei mehr als 20 Punkten erfolgreich absolviert.
Bitte drucken Sie diese drei Blätter einfach aus und bearbeiten Sie die Aufgaben auf dem Ausdruck.

Die Abgabe kann bis zum **22.5.2018**, 12.00 Uhr in dem mit SWT gekennzeichneten Briefkasten in der 1. Etage der OH12 am Übergang zum Erdgeschoß OH14 erfolgen.

Name/Matrikelnummer (Gruppenabgaben von bis zu 3 Studierenden):

- _____ / _____
- _____ / _____
- _____ / _____

Übungsgruppe: _____

Aufgabe 1 – Entwurfsmuster Dekorierer

(3 Punkte) Die Zuteilung von Aufgaben und die Bezahlung von Professoren einer Universität soll durch ein ***Dekorierer***-Muster modelliert werden, das die im Folgenden beschriebenen Eigenschaften geeignet umsetzt.

- Ein Professor ist eine Person mit einem Namen und einer Vergütung. Atom, extends Person
- Ein Professor kann Vorlesungen halten. Für jede Vorlesung wird ein Vorlesungsgeld gezahlt.
- Ein Professor kann Bachelorarbeiten betreuen. Für jede Betreuung wird ein Prüfungshonorar gezahlt.
- Ein Professor kann Bücher veröffentlichen, für die die Universität keine Zahlungen leistet. Ein Buch hat einen Titel.

Geben Sie ein geeignetes UML-Klassendiagramm für das Dekorierer-Muster an.

Skizzieren Sie für jede der von Ihnen vorgesehenen Klassen ***entweder*** als informale Beschreibung ***oder*** als Java-Programmcode die Implementierungen der Methoden `gehalt` und `vorlesung`.

a) Für das oben modellierte Dekorierer-Muster soll die Methode `int gehalt()` angelegt werden, die für einen mit Hilfe der Objekte des Dekorierer-Musters beschriebenen Professor dessen gesamte Bezahlung ermittelt.

```
public class Vorlesung
implements ProfComponent{
…

public int gehalt(){
   return next.gehalt() + this.entgeld;
}

…
}
```

```
public class BA
implements ProfComponent{
…

public int gehalt(){
   return next.gehalt() + this.entgeld;
}

…
}
```

```
public class BA
implements ProfComponent{
…

public int gehalt(){
   return next.gehalt() + this.entgeld;
}

…
}
```

```
public class Professor
implements ProfComponent{
… public int gehalt(){return 0;}…
}
```

```
public class Buch
implements ProfComponent{
… public int gehalt(){return next.gehalt();}…
}
```

Abgabe-Übungsblatt A2

b) Für das oben modellierte Dekorierer-Muster soll die Methode `boolean vorlesung()` angelegt werden, die für einen mit Hilfe der Objekte des Dekorierer-Musters beschriebenen Professor feststellt, ob mindestens eine Vorlesung gehalten wird.

analog mit:

return next.vorlesung() ll this.isVorlesung();

Aufgabe 2 – Entwurfsmuster Kompositum

(3 Punkte) In einem Speditionsauftrag werden verschiedene Versandgüter transportiert. Zeichnen Sie ein Klassendiagramm für ein ***Kompositum***-Muster, das die im Folgenden beschriebenen Beziehungen zwischen den beschriebenen Versandgütern geeignet umsetzt:

ArrayList

Es kommen die Versandgüter Paket, Versandbox, Karton und Kiste vor.
Ein Paket kann Versandboxen, Kartons, Kisten und andere Pakete enthalten.
Eine Versandbox kann Pakete, Kartons, Kisten und andere Versandboxen enthalten.
Jedes Versandgut besitzt ein Attribut `int gewicht` und ein Attribut `boolean brennbar`.

Skizzieren Sie für jede der von Ihnen vorgesehenen Klassen ***entweder*** als informale Beschreibung ***oder*** als Java-Programmcode die Implementierungen der Methoden `anzahlBoxen` und `gefahr`.

a) Für das oben modellierte Kompositum-Muster soll eine Methode `int anzahlBoxen()` angelegt werden. Die Methode `anzahlBoxen` soll für einen mit dem Kompositum-Muster modellierten Speditionsauftrag die Anzahl der enthaltenen Versandboxen liefern.

Abgabe-Übungsblatt A2

```
public class Paket extends PkgComp{
   private ArrayList<PkgComp> children;
   …
   public int anzahlBoxen(){
   int anzahl = 0;
   for(PkgComp c : children){
      anzahl += c.anzahlBoxen();
      return anzahl;
   }
}
```

```
public class Karton extends PkgComp{
   private ArrayList<PkgComp> children;
   …
   public int anzahlBoxen(){
      return 1;
   }
}
```

rest analog

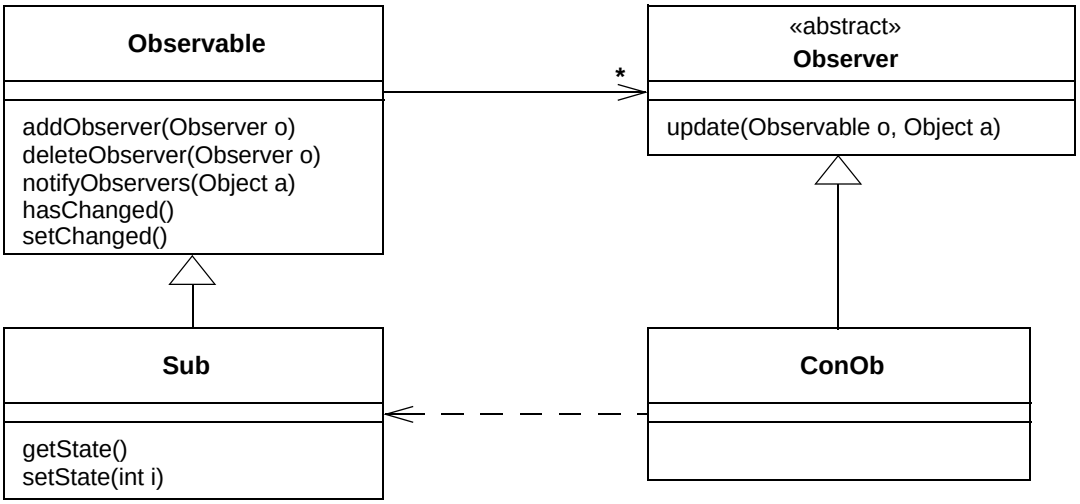
b) Für das oben modellierte Kompositum-Muster soll eine Methode `int gefahr()` angelegt werden. Die Methode `gefahr` soll für einen mit dem Kompositum-Muster modellierten Speditionsauftrag das Gewicht aller brennbaren Kartons und Kisten liefern.

```
public class Paket extends PkgComp{
   private ArrayList<PkgComp> children;
   …
   public int gefahr(){
   int gew = 0;
   for(PkgComp c : children){
      gew += c.gefahr();
   }
   if(this.brennbar)    gew += this.gewicht;
   return gew;
}
```

```
public class Karton extends PkgComp{
   private ArrayList<PkgComp> children;
   …
   public int anzahlBoxen(){
      return this.brennbar ? this.gewicht : 0;
   }
}
```

rest analog

Aufgabe 3 – Entwurfsmuster Beobachter



(2 Punkte) Das Klassendiagramm zeigt die aus der Vorlesung bekannte Umsetzung des Entwurfsmusters ***Beobachter*** in Java. Vervollständigen Sie das unten begonnene Sequenzdiagramm derart, dass es für den Aufruf der Methode `setState(5)` alle notwendigen Aufrufe der verschiedenen im Klassendiagramm vorgegebenen Methoden zeigt. Gehen Sie davon aus, dass die drei Objekte `co1, co2` und `co3` bereits bei `s` angemeldet sind.

Abgabe-Übungsblatt A2