


Klausur Softwaretechnologie SS 2014

Name:	
Vorname:	
Immatrikulationsnummer:	

Aufgabe	Maximale Punktzahl	Erreichte Punktzahl
1	6	
2	2	
3	14	
4	15	
5	(3+23+11+9)=46	
6	7	
Gesamt	90	

Hinweise:

- In der Klausur ist als Hilfsmittel lediglich ein **A4-Blatt, beidseitig beschrieben**, zugelassen.
- Die Klammerung der Aufgabenblätter darf **nicht** entfernt werden.
- Tragen Sie bitte die Lösungen auf den Aufgabenblättern ein!
- Verwenden Sie keine roten, grünen Stifte oder Bleistifte!
- Es ist kein eigenes Papier zu verwenden! Bei Bedarf ist zusätzliches Papier bei der Aufsicht erhältlich. Bitte jedes zusätzliche Blatt mit Name, Vorname und Immatrikulationsnummer beschriften.
- Es sind alle Aufgabenblätter abzugeben!
- Ergänzen Sie das Deckblatt mit Name, Vorname und Immatrikulationsnummer!
- Halten Sie Ihren Studentenausweis und einen Lichtbildausweis zur Identitätsprüfung bereit.
- **Achtung!** Das Zeichen  heißt: **Hier ist Java-Text einzufügen!**

Auftragsbearbeitung

Ein **System zur Auftragsbearbeitung** unterstützt alle administrativen Tätigkeiten, die für die Bearbeitung eines Kundenauftrages notwendig sind: Eine Auftragsbearbeitung wird ausgelöst, indem ein Kunde einen Auftrag über zu liefernde Artikel erteilt (Bestellung). Zunächst wird der Kunde identifiziert und seine Bonität geprüft. Dann muss die Verfügbarkeit der Artikel geprüft werden. Entweder muss ein Rückstellposten gebildet werden, oder es wird die Auslieferung der bestellten Artikel angewiesen. Der Artikelbestand wird dann aktualisiert. Wenn der Bestand zu niedrig ist, werden Artikel nachbestellt, d.h. beim Lieferanten bestellt. Zum Schluss wird die Rechnung erstellt.

Das zugehörige **Domänenmodell** ist auf der nächsten Seite (Seite 3) zu sehen.

Aus der Sicht des Systems gibt es zwei Klassen von externen Stakeholdern – die Kunden und die Lieferanten für die auszuliefernden Artikel.

Sowohl der **Kunde** als auch der **Lieferant** sind ähnliche und komplexe Objekte. Sie werden beide durch ihre Postadresse, Kontakte, unter denen sie erreichbar sind sowie durch einen Bonitätswert (sobald dieser bekannt ist) beschrieben. Für die Kunden werden Bestellungen bearbeitet (siehe oben). Für Lieferanten werden Lieferaufträge ausgelöst, um Artikel nachzubestellen.

Ein drittes zentrales Geschäftsobjekt sind die Artikel. Ein **Artikel** im Lagerbestand wird durch eine Artikelnummer, den Artikelnamen, den Artikelpreis, seine derzeit im Lager vorrätige Anzahl (Artikelbestand) und eine Mindestmenge beschrieben. Die Mindestmenge ist die Menge, die im Lager mindestens vorrätig sein sollte. Wenn diese erreicht ist, muss bei dem Lieferanten mit dem geringsten Lieferantenpreis nachbestellt werden.

Aufgabe 1: Rechnungsinformationen (6 Punkte)

Für jede Bestellung, die zumindest teilweise erfüllt wird, wird eine Rechnung erstellt.

Eine Rechnung beinhaltet neben den im Domänenmodell angegebenen Attributen:

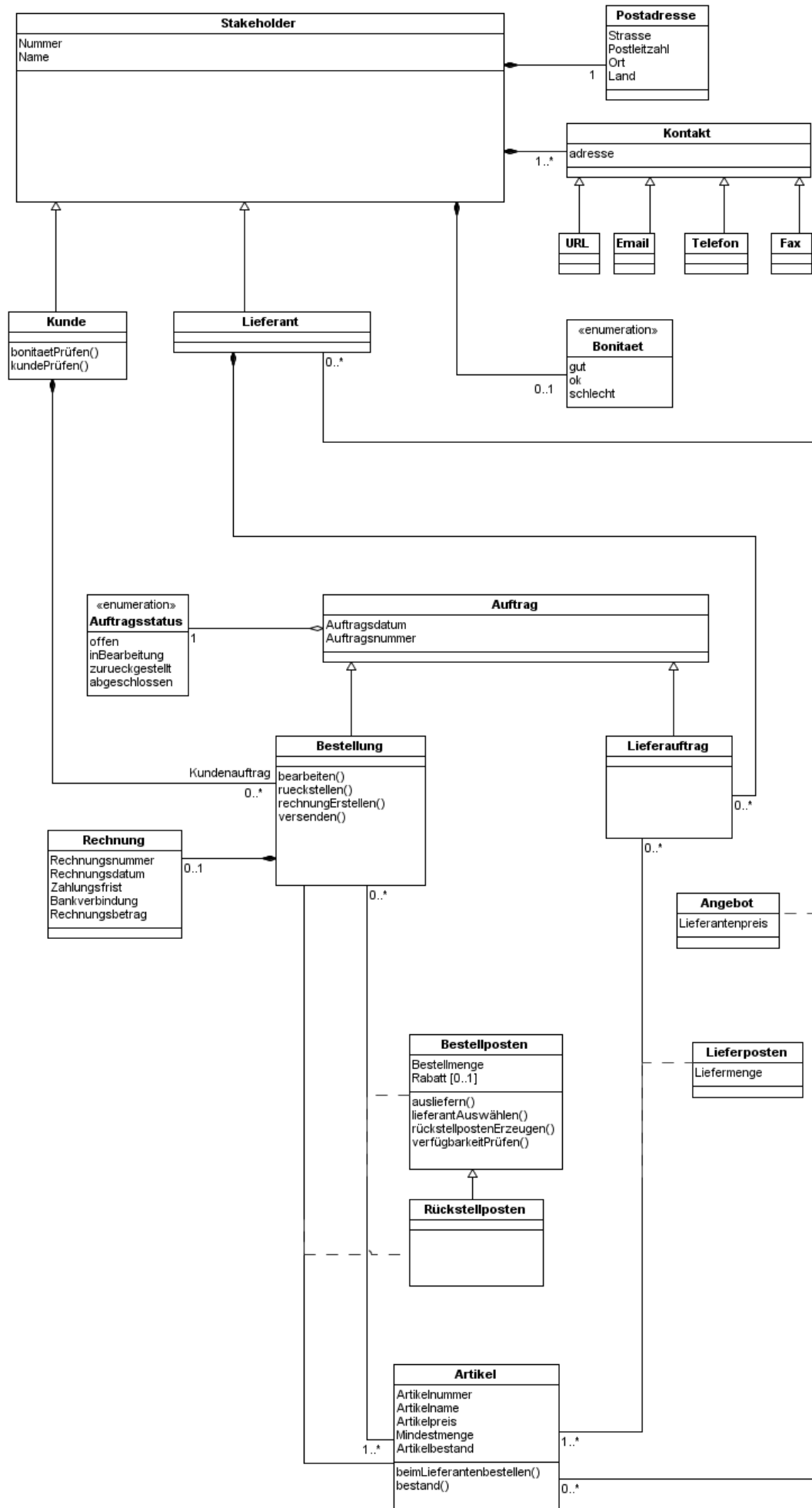
- Kundendaten
- Rechnungsposten mit Liefermenge pro Artikel, Artikelpreis pro Artikel, Postenpreis (Bestellmenge * Artikelpreis - Rabatt)
- Bankverbindung
- Rechnungsbetrag (Summe aller Postenpreise + Mehrwertsteuer)

Ergänzen Sie das Domänenmodell so, dass alle Informationen verfügbar sind, um die Rechnung zu erstellen!

Aufgabe 2: Versandauftrag (2 Punkte)

Für jeden Artikel, der bestellt wird und ausgeliefert werden kann, wird ein eigener Versandauftrag an die Lagerverwalter geschickt.

Ergänzen Sie das Domänenmodell um Versandaufträge!



Aufgabe 3: Anwendungsfalldiagramm (14 Punkte)

Betrachten wir nun das Gesamtsystem. Kunden können Kundenauskünfte abfragen und Bestellungen aufgeben. Lieferanten erhalten die Lieferaufträge und liefern neue Ware. Neben den Kunden und Lieferanten gibt es weitere Akteure, die mit dem System arbeiten. Lagerverwalter erhalten die Versandaufträge und verschicken die Ware. Buchhalter erhalten die Rechnungen, die sie prüfen und versenden. Auftragsverwalter überwachen die Rückstellungen und stoßen die Rückstellungsbearbeitung an. Nicht vergessen werden darf, dass sich jeder Akteur zunächst am System anmelden muss.

Erstellen Sie ein Anwendungsfalldiagramm! Verwenden Sie für die Benennung eines Anwendungsfalles ein geeignetes Verb (bzw. ein Substantiv und ein Verb).

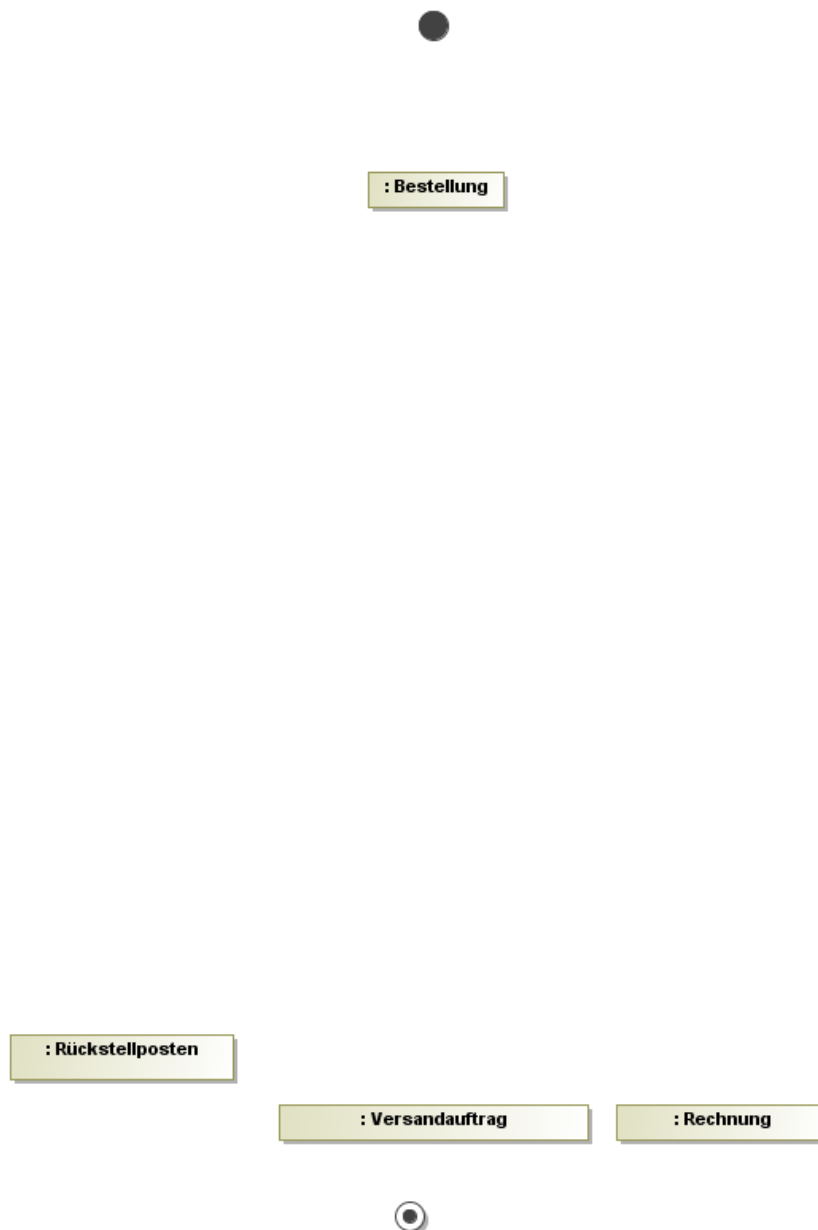
Aufgabe 4: Aktivitätsdiagramm für die Bearbeitung einer Bestellung (15 Punkte)

Bei der Bearbeitung einer Bestellung wird zunächst geprüft, ob der Kunde überhaupt bekannt ist. Wenn er bekannt ist, wird seine Bonität geprüft. Falls der Kunde unbekannt oder kreditunwürdig ist, wird die Bestellung abgelehnt, sonst werden die Bestellposten der Reihe nach abgearbeitet.

Bei der Bearbeitung eines Bestellpostens wird zuerst geprüft, ob der bestellte Artikel registriert ist. Wenn nicht, wird der Bestellposten abgewiesen. Wenn ja, wird geprüft, ob der Artikelbestand ausreichend ist, bzw. ob der Artikelbestand plus der Mindestmenge größer als die Bestellmenge ist. Falls die Menge nicht ausreichend ist, wird der Bestellposten zurückgestellt und ein Rückstellposten gebildet. Falls der Artikelbestand ausreichend ist, wird die Bestellmenge vom Artikelbestand abgezogen und ein Versandauftrag für den Bestellposten erstellt. Nach Abarbeitung aller Bestellposten wird eine Rechnung erstellt.

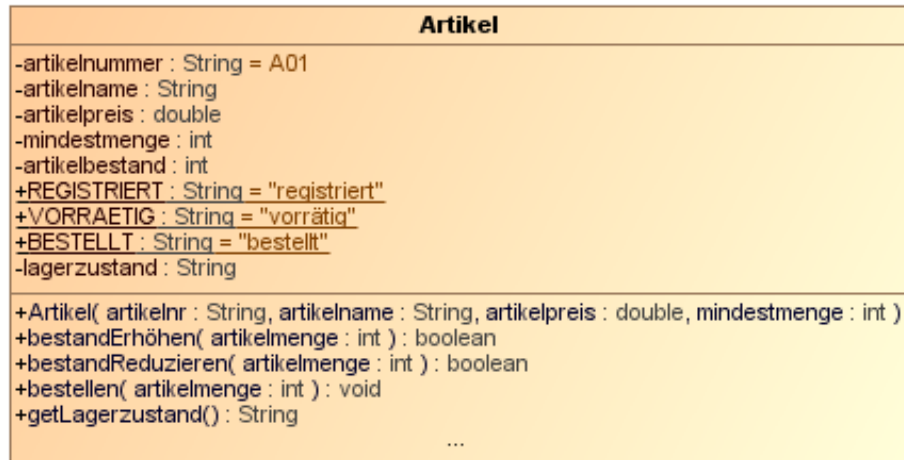
Modellieren Sie diesen Geschäftsprozess mit einem UML-Aktivitätsdiagramm!

Das Aktivitätsdiagramm ist nachfolgend unvollständig angegeben. Ergänzen Sie Aktionen, Entscheidungen sowie den Kontroll- und Objektfluss. Alle dazu notwendigen Objekte sind bereits im Diagramm vorhanden. Geben Sie den (durch Sie ergänzten) Modellelementen geeignete Namen.



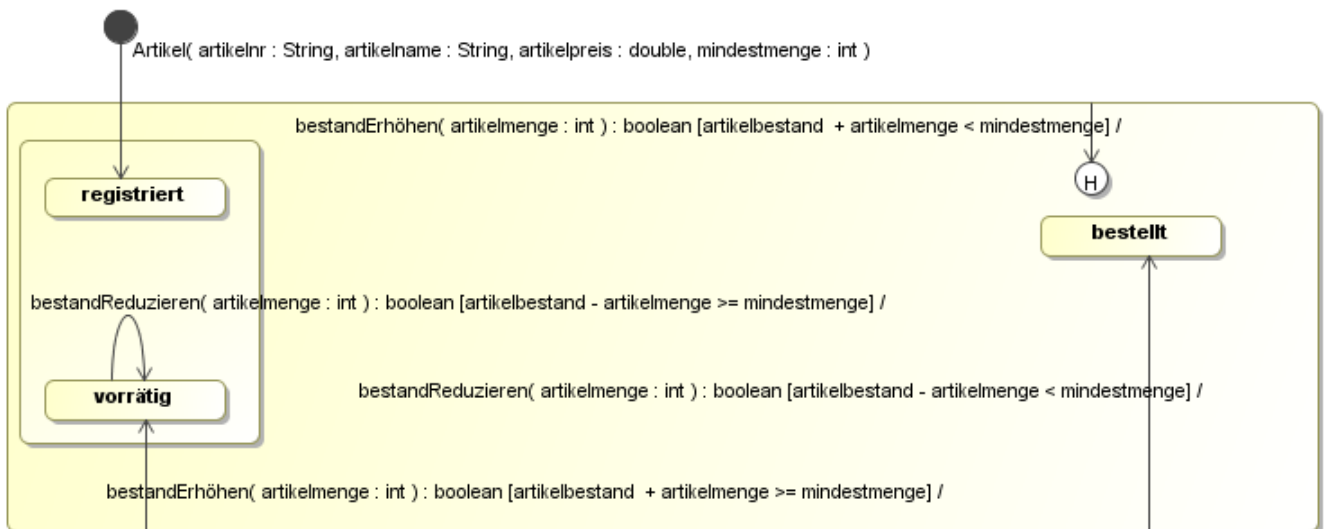
Aufgabe 5: Klasse Artikel (46 Punkte)

Die Klasse Artikel ist im Entwurf wie folgt verfeinert worden:



Jedes Artikelobjekt beschreibt sowohl den Artikel selbst als auch seinen Bestand im Lager (siehe Seite 2). Der Lebenszyklus eines Artikelobjektes kann wie folgt beschrieben werden und ist in dem unten stehenden Zustandsdiagramm (Verhaltenszustandsmaschine) modelliert. Dabei wurden Aktionen nicht modelliert.

Zu Beginn wird das Artikelobjekt erzeugt. Damit ist der Artikel im Bestand registriert, aber noch mit einem Artikelbestand gleich null (0). Der Artikelbestand kann jederzeit erhöht werden (`bestandErhöhen()`). Das kann aber nur dann geschehen, wenn danach der Artikelbestand gleich oder größer der geforderten Mindestmenge ist. Danach ist der Artikel im Bestand vorrätig. Mit der Methode `bestandReduzieren()` wird bei einer (Kunden-)Bestellung die Anzahl der bestellten Artikel („Bestellmenge“) aus dem Lagerbestand genommen. Der Artikelbestand wird entsprechend reduziert. Falls der Artikel derzeit nicht vorrätig ist oder dies nach der Entnahme des Artikels aus dem Lagerbestand gelten würde, bleibt der Artikelbestand unangetastet (d.h. der Artikel wird nicht ausgeliefert), und es muss nachbestellt werden (`bestellen(artikelmenge)`).



a) Ergänzen Sie in Java-Notation fehlende Aktionen im Zustandsdiagramm (3 Punkte)!

b) Implementieren Sie die Klasse Artikel (23 Punkte)!

- Verwenden Sie dazu NICHT das State-Pattern!
- Falls sich der Artikelbestand nicht verändert, geben die Methoden `bestandErhoehen()` und `bestandReduzieren()` `false` aus, ansonsten `true`.
- Die Methode `bestellen()` soll (in der Klausur) als „leere“ Methode implementiert werden.



- c) Geben Sie für jeden Zustandsübergang im Lebenszyklus eines Artikel-Objektes genau einen passenden Testfall an! Ergänzen Sie dazu die folgenden Testfalltabellen und beachten Sie, dass Sie die Oberzustände im Zustandsmodell auflösen müssen (11 Punkte).

- Gehen Sie von einem Artikelobjekt aus, welches wie folgt erzeugt wird:

```
Artikel a = new Artikel ("A01", "Galaxy_S6", 599.50, 20);
```

- Berücksichtigen Sie nur zulässige Parameter (artikelmenge > 0)

Testfalltabelle für die Methode `bestandErhoehen(int artikelmenge)`

Testfall-nummer	Attribute des Artikelobjektes			Parameter	lagerzustand (expected)
	lagerzustand	artikelbestand	mindestmenge	artikelmenge	
1					
2					
3					
4					
5					

Testfalltabelle für die Methode `bestandReduzieren(int artikelmenge)`

Testfall-nummer	Attribute des Artikelobjektes			Parameter	lagerzustand (expected)
	lagerzustand	artikelbestand	mindestmenge	artikelmenge	
6					
7					
8					
9					
10					
11					

- d) Implementieren Sie prototypisch eine Testklasse für die Klasse Artikel entsprechend Ihrer Testfalltabellen aus Teilaufgabe c) (9 Punkte)!

- Ergänzen Sie dazu den folgenden Java-Code! Beschränken Sie sich auf einen (junit 3.8.1) Testfall pro Testfalltabelle.
- Tragen Sie in den Kommentar zu jeder Testfallmethode die zugehörige Testfallnummer aus Ihren Testfalltabellen ein!

- Nehmen Sie an, dass die Klasse Artikel zwei zusätzliche Hilfsmethoden besitzt:

```
public void setArtikelbestand(int artikelbestand) {
    this.artikelbestand = artikelbestand;
}

public void setLagerzustand(String lagerzustand) {
    this.lagerzustand = lagerzustand;
}
```

```
import junit.framework.TestCase; // Version 3.8.1
import junit.swingui.TestRunner;

public class ArtikelTest extends TestCase {

    private Artikel a;

    protected void setUp() {
        a = new Artikel(1, "Brot", 1.5, "Brot");
    }

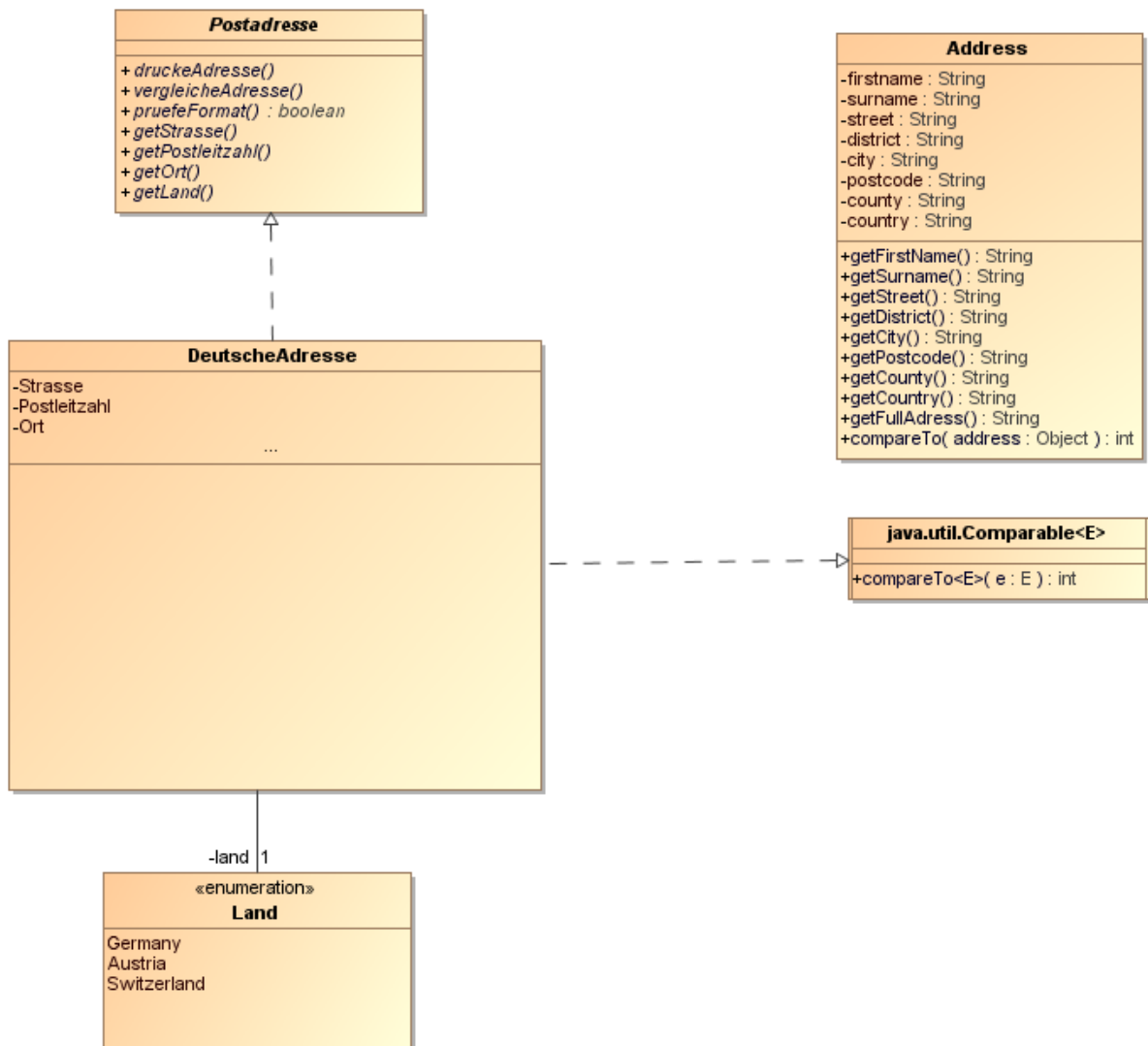
    public void testBestandErhoehen() { // Testfallnummer: ...
        a.setArtikelbestand(a.getArtikelbestand() + 1);
        assertEquals("Artikelbestand sollte um 1 erhöht werden",
            a.getArtikelbestand(), 2);
    }

    public void testBestandReduzieren(){ // Testfallnummer: ...
        a.setArtikelbestand(a.getArtikelbestand() - 1);
        assertEquals("Artikelbestand sollte um 1 reduziert werden",
            a.getArtikelbestand(), 0);
    }

}
```

Aufgabe 6: Postadresse (7 Punkte)

Die Klasse Postadresse wird im Entwurf wie folgt verfeinert:



Die im Domänenmodell gegebene Klasse Postadresse wird als *Interface* definiert, welches durch die Klasse DeutscheAdresse implementiert wird. Zur Implementierung soll ein **Klassenadapter** genutzt werden. Die dabei wiederverwendete Klasse ist die Klasse Address.

- Ergänzen Sie das Entwurfsklassendiagramm um folgende Elemente:
 - Methoden der Klasse DeutscheAdresse
 - Beziehung, die durch den Klassenadapter implementiert werden muss
- Tragen Sie das Entwurfsmuster Klassenadapter in UML-Notation in das Entwurfsklassendiagramm ein!