Übung 8, Lösungsvorschlag

TECHNISCHE UNIVERSITÄT DARMSTADT

Aufgabe 1 (Wissensfrage)

a) Welche Art von Beziehungen wird durch (a) Attribute und (b) Vererbung abgebildet? Nennen Sie jeweils ein Beispiel.

b) Welche Vorteile ergeben sich durch Vererbung bei der objektorientierten Programmierung?

c) Was wird bei der Vererbung von der Oberklasse auf die Subklasse übertragen? Gibt es Elemente der Oberklasse, die nicht auf die Subklasse übertragen werden?

d) Was sind überschriebene Methoden und wie werden sie implementiert?

e) Was ist Ihre Meinung zu folgender Aussage: Java ist eine typisierte Sprache. Daher können Objekte nur in solchen Variablen gespeichert werden, bei deren Deklaration genau der Typ des Objekts angegeben wurde.

Lösung Aufgabe 1

a) Attribute bilden „hat-Beziehungen“ ab, z.B. „Ein Tisch hat vier Beine“.
Vererbung bildet „ist-Beziehungen“ ab, z.B. „Studenten und Professoren sind Personen“.

b) - Wiederverwendung (Vermeidung von Redundanz und Fehlervermeidung)
- Schrittweise Entwicklung vom Allgemeinen zum Speziellen
- konsistente Schnittstellen

c) Es werden alle Attribute und Methoden geerbt, allerdings keine Konstruktoren!

> **Hinweis:** In den weiteren Vorlesungen werden wir noch auf das Thema „Sichtbarkeit von Attributen sowie Methoden“ eingehen: Es werden ausschließlich Attribute und Methoden geerbt, die den Sichtbarkeits-Modifier **public** oder **protected** haben – oder keine Sichtbarkeitsangabe (package), sofern sich Oberklasse und Subklasse im selben Paket befinden. Hieraus ergibt sich, dass **private** Attribute sowie Methoden der Oberklassen nicht an die Subklassen vererbt werden.

d) Subklassen können geerbte Methoden neu implementieren. Dieser Vorgang wird als Überschreiben bezeichnet. Hierzu wird in der Subklasse eine Methode mit derselben Signatur implementiert, sie überschreibt die entsprechende Methode aus der Oberklasse. Beim Aufruf einer überschriebenen Methode auf (Objekten) der Subklasse wird die neue Implementierung der überschreibenden Methode aufgerufen. Die Subklasse kann auf die überschriebenen Methoden der Oberklasse über die Referenz super zugreifen.

e) Die Aussage ist falsch. Java ist zwar eine typisierte Sprache, die Vererbungsbeziehungen erlauben es jedoch, Objekte in Variablen des eignen und jedes übergeordneten Typs zu speichern. Man unterscheidet hier zwischen dem statischen und dem dynamischen Datentypen. Der statische Datentyp ist der Typ der Variablen, der dynamische der des Objekts.

Aufgabe 2 (Kochbuch)

a) Erstellen Sie eine Klasse `Book`, die das Objektattribut `title` vom Datentyp `String` und das ganzzahlige Objektattribut `pagecount` enthält.

b) Erstellen Sie in der Klasse `Book` eine Methode `toString()` mit dem Rückgabewert `String`, die den Buchtitel und die Seitenzahl zurückgibt.

c) Erstellen Sie nun eine neue Klasse `SchoolBook`. Diese soll von der Klasse `Book` erben und ist damit Unterklasse von `Book`.

d) Geben Sie der Unterklasse `SchoolBook` ein ganzzahliges Objektattribut `gradeLevel`. Das Attribut soll mit dem Wert `-1` initialisiert werden.

e) Erstellen Sie nun eine weitere Klasse `CookBook`, die ebenso von der Klasse `Book` erbt und damit zur Unterklasse wird.

f) Geben Sie der Klasse `CookBook` ein Objektattribut `vegetarian` vom Typ `boolean`. Implementieren Sie außerdem einen Konstruktor, der einen Parameter enthält, durch den das Attribut `vegetarian` mit dem übergebenen Wert belegt wird. Außerdem soll eine Objektmethode mit dem Bezeichner `isVegetarian` erstellt werden, die `true` zurückgibt, wenn das Kochbuch vegetarisch ist und `false`, wenn nicht.

g) Überschreiben Sie nun innerhalb der Klasse `CookBook` die Methode `toString()`. Die überschreibende Methode soll wiederum einen String zurückgeben. Dieser soll aus dem Text der `toString()`-Methode aus der Oberklasse bestehen, ergänzt um die Angabe, ob das Kochbuch vegetarisch ist oder nicht. Verwenden Sie hierzu einen Verweis auf die Methode der Oberklasse, anstatt den String wiederholt zu schreiben.

h) Wie lautet die Ausgabe der folgenden `main`-Methode?

```
1     public class BookManagementSystem {
2           public static void main(String[] args) {
3                 SchoolBook myComputerBook = new SchoolBook();
4                 myComputerBook.title = "Java ist auch eine Insel";
5                 myComputerBook.pageCount = 1308;
6                 myComputerBook.gradeLevel = 42;
7                 System.out.println(myComputerBook);
8
9                 CookBook myCookBook = new CookBook(true);
10                myCookBook.title = "Vegetarisch Kochen für
                  Fortgeschrittene";
11                myCookBook.pageCount = 84;
12                System.out.println(myCookBook);
13
14                Book myBook = myCookBook;
15                System.out.println(myBook);
16          }
17    }
```

Lösung Aufgabe 2

```java
public class Book {
	public String title;
	public int pageCount;

	public Book() {

	}

	public String toString() {
		return title + ", " + pageCount + " S.";
	}
}
```

```java
public class Schoolbook extends Book {
	public int gradeLevel = -1;

	public Schoolbook() {
		super();
	}
}
```

```java
public class CookBook extends Book {
	public boolean vegetarian;

	public CookBook(boolean veg) {
		super();
		vegetarian = veg;
	}

	public boolean isVegetarian() {
		return vegetarian;
	}

	public String toString() {
		if (vegetarian) {
			return super.toString() + " Vegetarisches Kochbuch.";
		} else {
			return super.toString() + " Kein vegetarisches
			Kochbuch.";
		}
	}
}
```

Die Ausgabe lautet:

```
Java ist auch eine Insel, 1308 S.
Vegetarisch Kochen für Fortgeschrittene, 84 S. Vegetarisches Kochbuch.
Vegetarisch Kochen für Fortgeschrittene, 84 S. Vegetarisches Kochbuch.
```

Aufgabe 3 (Programmieren – Verwaltung von Sparkonten)

***Hinweis**: Der Programmcode der Musterlösung zur Sparkonto-Aufgabe aus Übung 6 kann in Moodle heruntergeladen werden.*

a) In den vorherigen Übungen haben Sie bereits ein Programm zur Verwaltung von Konten und Sparkonten implementiert. Nachdem Sie sich mit dem Konzept der Vererbung auseinandergesetzt haben, fällt Ihnen auf, dass Ihre Lösung noch nicht optimal ist. Um Zukünftig einfacher neue Typen von Konten hinzufügen zu können, möchten Sie alle speziellen Kontotypen von der Klasse `Konto` erben lassen.

Alle `Konto`-Objekte sollen über die Objektattribute `kontonummer` und `kontostand` verfügen. Das Attribut `kontonummer` soll bei der Erzeugung eines `Konto`-Objekts mit einem dem Konstruktor übergebenen Wert belegt werden, während der `kontostand` während der Erzeugung mit 0 initialisiert wird.

Die Klasse Konto soll außerdem drei Methoden definieren:

- `public boolean einzahlen(double)` – bereits in vorheriger Übung implementiert
- `public boolean auszahlen(double)` – bereits in vorheriger Übung implementiert
- `public void druckeKontoauszug()`

Zur Erinnerung: Die Methoden `einzahlen(double)` und `auszahlen(double)` erhalten jeweils den Betrag als Parameter übergeben und liefern einen Wert zurück, der angibt, ob die Transaktion ausgeführt wurde oder nicht. Diese Transaktionen können nicht ausgeführt werden, falls der übergebene Parameter einen negativen Wert hat. Eine Auszahlung kann nur erfolgen, wenn der Auszahlungsbetrag nicht größer ist als der Kontostand.

Die Methode `druckeKontoauszug()` soll Angaben über Kontonummer und Kontostand auf der Konsole ausgeben.

b) Die bereits existierende Klasse `Sparkonto` (vorherige Übung) soll nun von der Klasse `Konto` erben, sodass nur das Attribut `zinssatz` sowie die Methoden `verzinsen` und `jahresabschluss` in der Klasse `Sparkonto` implementiert sind.

c) Mittlerweile hat ihr Auftraggeber entdeckt, dass sich mit Dispozinsen sehr gutes Geld verdienen lässt. Dazu möchte die Bank nun auch ein `Girokonto` anbieten. Die Klasse `Girokonto` verfügt über ein Attribut `dispoLimit` vom Datentyp `double`, das bei der Erzeugung eines Girokontos mit einem Wert belegt werden soll. Überschreiben Sie die Methode `auszahlen(double)` in der Klasse `Girokonto`, sodass auch negative Kontostände bis zum Erreichen des Dispolimits möglich sind.
***Hinweis:** Die Methode `auszahlen(double)` ist bereits durch die Ober-Klasse `Konto` implementiert, welche überprüft, dass das Konto ausreichend gedeckt ist.*

d) Außer diesen drei Klassen benötigen Sie ein Hauptprogramm (`main`-Methode) in der Klasse `Bank`. Zunächst soll der Zinssatz für Sparkonten auf 0,6% (0,006) gesetzt werden. Danach werden zwei Girokonten und ein Sparkonto erzeugt, die alle in einem Array gespeichert werden sollen. Die Werte für Kontonummer und Dispolimit (bei Girokonten) können Sie nach Belieben initialisieren. Auf diesen Konten sollen die folgenden Transaktionen ausgeführt werden. Nach jeder Transaktion soll für das betroffene Konto ein Kontoauszug gedruckt werden.
***Hinweis**: Machen Sie sich klar, unter welchen Umständen die Methode `verzinsen()`, auch auf einem Objekt der Klasse `Konto` (z.B. dem Konto „Spar“) aufgerufen werden kann.*

Konto	Einzahlung	Auszahlung
Giro1	1000	
Giro1		1500
Giro2	200	
Giro2		700
Spar	5000	
Spar		verzinsen()

Lösung Aufgabe 3

```
 1  public class Konto {
 2        public int kontonummer;
 3        public double kontostand;
 4
 5        public Konto(int nr) {
 6              kontonummer = nr;
 7              kontostand = 0.0;
 8        }
 9
10        public boolean einzahlen(double betrag) {
11              if (betrag > 0) {
12                    kontostand = kontostand + betrag;
13                    return true;
14              } else {
15                    return false;
16              }
17        }
18
19        public boolean auszahlen(double betrag) {
20              if (betrag > 0 && kontostand >= betrag) {
21                    kontostand = kontostand - betrag;
22                    return true;
23              } else {
24                    return false;
25              }
26        }
27
28        public void druckeKontoauszug() {
29              System.out.println("Konto " + kontonummer + ": " +
    kontostand);
30        }
31  }
```

```java
public class Sparkonto extends Konto {
	static double zinssatz;

	public Sparkonto(int nr) {
		super(nr);
	}

	public void verzinsen() {
		double zinsen = kontostand * zinssatz;
		einzahlen(zinsen);
	}

	public static void jahresabschluss(Sparkonto[] konten) {
		for (int i = 0; i < konten.length; i++) {
			konten[i].verzinsen();
		}
	}
}
```

```java
public class Girokonto extends Konto {
	public double dispoLimit;

	public Girokonto(int nr, double dispo) {
		super(nr);
		dispoLimit = dispo;
	}

	public boolean auszahlen(double betrag) {
		if (betrag > 0 && (kontostand + dispoLimit) >= betrag) {
			kontostand = kontostand - betrag;
			return true;
		} else {
			return false;
		}
	}
}
```

```java
public class Bank {
	public static void main(String[] args) {

		Sparkonto.zinssatz = 0.006;
		Konto[] konten = new Konto[3];

		konten[0] = new Girokonto(1, 300);
		konten[1] = new Girokonto(2, 1000);
		konten[2] = new Sparkonto(3);

		konten[0].einzahlen(1000);
		konten[0].druckeKontoauszug();
		konten[0].auszahlen(1500);
		konten[0].druckeKontoauszug();
		konten[1].einzahlen(200);
		konten[1].druckeKontoauszug();
		konten[1].auszahlen(700);
		konten[1].druckeKontoauszug();
		konten[2].einzahlen(5000);
		konten[2].druckeKontoauszug();

		if (konten[2] instanceof Sparkonto) {
			Sparkonto s = (Sparkonto) konten[2];
			s.verzinsen();
			s.druckeKontoauszug();
			// Alternativ funktioniert auch
			// ((Sparkonto)konten[2]).verzinsen();
		}
	}
}
```