



Programmieren: Einstieg in die Informatik mit Java

WS 2006/2007

Dr. G. Bohlender
Dipl.–Math. techn. M. Richter

15.01.2007

Aufgabenblatt 10

Bearbeitungszeitraum: 17.01.2007 – 31.01.2007

Aufgabe 24: *Polymorphismus*

Beim objektorientierten Programmieren ist es möglich, von einer bestehenden Objektklasse, der sogenannten *Superklasse*, eine sogenannte *Subklasse* abzuleiten, welche als Variante der Superklasse verstanden werden kann. Unter *Polymorphismus* versteht man nun, dass man eine Instanzmethode, die in der Superklasse definiert wurde, in der Subklasse *überschreiben* kann. Auf diese Weise entsteht der Eindruck, dass diese Instanzmethode unterschiedliche Funktionalitäten besitzt, je nachdem, ob sie für die Super- oder für die Subklasse aufgerufen wird. In dieser Aufgabe soll das Konzept des Polymorphismus anhand eines einfachen Beispiels verdeutlicht werden.

- (a) Erstellen Sie eine Objektklasse mit dem Namen `Vogel`. Diese Klasse soll als einzigen Bestandteil eine Instanzmethode mit dem Namen `singe()` besitzen, welche nichts tut und daher auch keinen Rückgabewert besitzt. Da Sie für die Objektklasse `Vogel` auch keinen Konstruktor definieren, besitzt diese lediglich den Standard-Konstruktor, d.h. Sie können mit dem Befehl `Vogel vogel1 = new Vogel();` ein Objekt vom Typ `Vogel` erzeugen und in der Variable `vogel1` speichern.
- (b) Leiten Sie von der Objektklasse `Vogel` eine Subklasse mit dem Namen `Spatz` ab. Erstellen Sie dazu eine Datei mit dem Namen `Spatz.java`, in der Sie folgenden Quelltext speichern:

```
public class Spatz extends Vogel { }
```

Auf diese Weise wird die Objektklasse `Spatz` zu einer Variante der Objektklasse `Vogel`. Jeder Variable vom Typ `Vogel` kann daher auch ein Objekt der Klasse `Spatz` zugewiesen werden. Der Aufruf `Vogel vogel2 = new Spatz();` erzeugt beispielsweise mittels Standard-Konstruktor ein Objekt der Klasse `Spatz` und speichert es in der Variable `vogel2` vom Typ `Vogel`.

- (c) Die Objektklasse `Spatz` besitzt als Subklasse der Superklasse `Vogel` ebenfalls die Instanzmethode `singe()`. Der Aufruf von `vogel2.singe()` bewirkt jedoch bislang nichts. Dies soll nun geändert werden. Überschreiben Sie dazu in der Objektklasse `Spatz` die Instanzmethode `singe()` derart, dass die Textzeile `Tschirp, Tschirp!` auf dem Bildschirm ausgegeben wird. Dies geschieht, indem Sie die Methode in der Objektklasse `Spatz` einfach neu definieren. Ihre Klassendefinition sollte also wie folgt aussehen:

```

public class Spatz extends Vogel {
    public void singe() {
        System.out.println("Tschirp, Tschirp!");
    }
}

```

- (d) Leiten Sie von der Objektklasse `Vogel` eine weitere Objektklasse mit dem Namen `Rabe` ab. Überschreiben Sie in dieser neuen Objektklasse die Instanzmethode `singe()` derart, dass `Krah, Krah!` auf dem Bildschirm ausgegeben wird.
- (e) Erstellen Sie ein Java-Programm mit dem Namen `Vogelrufe`. Definieren Sie in diesem Programm zwei Variablen vom Typ `Vogel`. Weisen Sie einer der beiden Variablen ein Objekt der Klasse `Spatz`, der anderen ein Objekt der Klasse `Rabe` zu. Rufen Sie danach für beide Variablen die Instanzmethode `singe()` auf. Der Polymorphismus besteht nun darin, dass der Aufruf dieser Methode einmal `Tschirp, Tschirp!` und einmal `Krah, Krah!` auf dem Bildschirm ausgibt.

Aufgabe 25: Vererbung

Unter *Vererbung* versteht man, dass eine Objektklasse, die von einer Superklasse abgeleitet wurde, bestimmte Methoden und Variablen der Superklasse übernimmt. Man sagt, die Superklasse *vererbt* gewisse Methoden und Variablen auf die Subklasse. Das Konzept der Vererbung soll anhand dieser Aufgabe verdeutlicht werden. Grundlage für diese Aufgabe ist das Programm, das in Aufgabe 22 entwickelt wurde.

- (a) Betrachten Sie noch einmal das Java-Programm `Personen` aus Aufgabe 22. In diesem Programm wurde eine Objektklasse mit dem Namen `Person` definiert, mit der eine Person, die einen Vornamen und einen Nachnamen besitzt, repräsentiert werden kann. Leiten Sie von dieser Klasse eine neue Objektklasse mit dem Namen `Akademiker` ab.
- (b) Die Objektklasse `Akademiker` besitzt alle Instanzvariablen und alle Instanzmethoden der Objektklasse `Person`. Zusätzlich soll sie eine weitere Instanzvariable vom Typ `String` mit dem Namen `titel` besitzen. Dies geschieht, indem man diese Instanzvariable in der Klasse `Akademiker` definiert. Ihre Klassendefinition sollte also wie folgt aussehen:

```

public class Akademiker extends Person {
    String titel;
}

```

- (c) Für die Objektklasse `Akademiker` muss ein eigener Konstruktor definiert werden. Definieren Sie für diesen Konstruktor drei formale Argumente vom Typ `String` mit den Namen `titel`, `vorname` und `nachname`. Der Konstruktor soll den Wert des formalen Arguments `titel` in der gleichnamigen Instanzvariable speichern. Außerdem muss der Konstruktor der Superklasse `Person` aufgerufen werden. Dies geschieht durch das Schlüsselwort `super`. Ihre Klassendefinition sollte also wie folgt aussehen.

```

public class Akademiker extends Person {
    String titel;

    public Akademiker(String titel, String vorname, String nachname) {
        super(vorname,nachname);
        this.titel = titel;
    }
}

```

- (d) Überschreiben Sie in der Klasse `Akademiker` die Instanzmethode `vollerName()` derart, dass zusätzlich zu dem Vor- und dem Nachnamen des Akademikers auch noch sein Titel zurück gegeben wird. Innerhalb dieser Methode können Sie die gleichnamige Instanzmethode der Klasse `Person` durch den Befehl `super.vollerName()` aufrufen.
- (e) Erstellen Sie ein Java-Programm mit dem Namen `Personen2`. Definieren Sie in diesem Programm zwei Variablen vom Typ `Person`. Weisen Sie einer der beiden Variablen eine `Person` mit dem Namen „Adam Riese“, der anderen einen `Akademiker` mit dem Namen „Dr. Eva Zwerg“ zu. Geben Sie jeweils den vollen Namen sowie die Namensinitialen auf dem Bildschirm aus.

Aufgabe 26 (Pflichtaufgabe): *Finanzplanung II*

Betrachten Sie noch einmal das Programm `Finanzplanung` aus Pflichtaufgabe 23. Dort wurde eine Objektklasse `Geldanlage` definiert, mit der eine Geldanlage repräsentiert werden kann. Diese Klasse besitzt Instanzvariablen, mit denen das Anfangsjahr t_0 , das Endjahr t_{Ende} , der angelegte Betrag B sowie der Zinssatz z der Geldanlage gespeichert werden konnte. Ferner besitzt diese Klasse eine Instanzmethode mit dem Namen `auszahlung()`, die für ein gegebenes Jahr t den Wert der Auszahlungsfunktion $a(t)$ zurück gibt.

- (a) Die Objektklasse `Geldanlage` beschreibt in der vorliegenden Form ein sogenanntes *Festgeld* zum Zinssatz z . Wir wollen nun weitere Formen der Geldanlage durch geeignete Objektklassen repräsentieren. Leiten Sie zunächst von der Objektklasse `Geldanlage` eine weitere Objektklasse mit dem Namen `Anleihe` ab.
- (b) Definieren Sie einen Konstruktor der Objektklasse `Anleihe` so, dass diesem ebenfalls Werte für t_0 , t_{Ende} , B und z übergeben werden können. Rufen Sie innerhalb des Konstruktors den Konstruktor der Superklasse `Geldanlage` mit diesen Argumenten auf. Verwenden Sie dazu das Schlüsselwort `super` wie im Aufgabenteil (c) der Aufgabe 25 beschrieben.
- (c) Überschreiben Sie in der Objektklasse `Anleihe` die Instanzmethode `auszahlung()` derart, dass sie folgende Auszahlungsfunktion repräsentiert:

$$a(t) = \begin{cases} -B & \text{falls } t = t_0 \\ zB & \text{falls } t_0 < t < t_{\text{Ende}} \\ (1+z)B & \text{falls } t = t_{\text{Ende}} \\ 0 & \text{sonst} \end{cases} \quad (1)$$

Hinweise zum Überschreiben von Instanzmethoden finden Sie im Aufgabenteil (c) von Aufgabe 24.

- (d) Leiten Sie von der Objektklasse `Geldanlage` eine weitere Objektklasse mit dem Namen `Sparvertrag` ab. Ein `Sparvertrag` besitzt folgende Auszahlungsfunktion:

$$a(t) = \begin{cases} -B & \text{falls } t_0 \leq t < t_{\text{Ende}} \\ \sum_{k=1}^{t_{\text{Ende}}-t_0} (1+z)^k B & \text{falls } t = t_{\text{Ende}} \\ 0 & \text{sonst} \end{cases} \quad (2)$$

Definieren Sie für die Klasse `Sparvertrag` einen Konstruktor und überschreiben Sie die Instanzmethode `auszahlung()` in geeigneter Weise.

- (e) Erstellen Sie ein Java-Programm mit dem Namen `Finanzplanung2`, in dem Sie analog zur Pflichtaufgabe 23 folgendes Portfolio simulieren:

Anlageform	Anfangsjahr	Endjahr	Betrag	Zinssatz
Festgeld	2	5	100	3.0 %
Anleihe	3	6	500	2.0 %
Sparvertrag	7	9	50	4.0 %

Sie können den Quelltext des Programm `Finanzplanung` weitestgehend übernehmen. Den Komponenten des Feldes `portfolio` weisen Sie einfach entsprechende Objekte der Klassen `Geldanlage`, `Anleihe` und `Sparvertrag` zu. Sie sollten folgende Ausgabe erhalten

Jahr	Summe der Auszahlungen
1	0.00
2	-100.00
3	-500.00
4	10.00
5	119.27
6	510.00
7	-50.00
8	-50.00
9	106.08
10	0.00