Übung 6, Lösungsvorschlag

TECHNISCHE UNIVERSITÄT DARMSTADT

Aufgabe 1 (Wissensfrage)

a) Erklären Sie den Unterschied zwischen Klassen- und Objektattributen.
   a. Wie äußert sich der Wert des Attributs zwischen verschiedenen Objekten?
   b. Wie werden diese deklariert?
   c. Wie wird auf diese zugegriffen?

b) Ist die Methode `double sqrt(double x)`, wie sie innerhalb der Klasse `Math` deklariert ist, eine Klassenmethode oder eine Objektmethode?

c) Wenn eine Objektvariable durch eine lokale Variable in einem Anweisungsblock überdeckt wird, ist es dann trotzdem möglich auf die überdeckte Variable zuzugreifen? Wenn ja, wie? Wenn nein, weshalb nicht?

d) Trifft dies auch auf überdeckte Klassenvariablen zu?

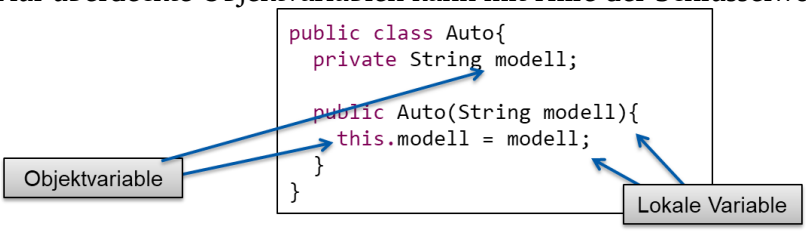
Lösung Aufgabe 1

a) Ein **Klassenattribut** gehört zu der jeweiligen Klasse, somit: Alle Instanzen der Klasse teilen sich dieselbe Variable für das Attribut (Folge: der Wert des Attributs ist in allen Instanzen gleich) Deklaration durch Verwendung des Modifiers static:
`static int counter = 42;`
Zugriff auf Klassenattribute erfolgt über den Klassennamen (kann innerhalb derselben Klasse weggelassen werden):
`int value = MyClass.counter;` (hier wird auf das Attribut `counter` der Klasse `MyClass` zugegriffen)

Ein **Objektattribut** gehört zu einer konkreten Instanz, somit: Jede Instanz verfügt über eine eigene Variable für das Attribut
Der Wert des Attributs kann in jeder Instanz verschieden sein. Deklaration durch Weglassen des Modifiers static: `int counter = 42;`
Zugriff erfolgt über den Namen der Objektvariablen (kann innerhalb derselben Klasse weggelassen werden):
`Auto a = new Auto("VW Golf");`
`String s = a.modell;` (hier wird auf das Attribut `modell` des in a gespeicherten `Auto`-Objektes zugegriffen)

b) Zahlreiche mathematische Methoden werden durch die Klasse `Math` bereitgestellt. Es handelt sich hierbei bei allen Methoden um Klassenmethoden, da diese direkt über die Klasse Math aufgerufen werden. Z.B.: `Math.sqrt(16);`
Dies ist sinnvoll, da diese Methoden ausschließlich Werte über Parameter übergeben bekommen.

c) Auf überdeckte Objektvariablen kann mit Hilfe der Schlüsselworts `this` zugegriffen werden.



d) Auf überdeckte Klassenvariablen kann ebenfalls zugegriffen werden. Da diese jedoch unabhängig von dem eigentlichen Objekt (this) sind, muss hierfür über den Klassennamen zugegriffen werden. (z.B. `Fahrer.beschleunigungsfaktor`).

Aufgabe 2 (Freunde) mit Lösungsvorschlag

Gegeben ist ein Programm mit den folgenden beiden Klassen `Info` und `Friend`:

```
1  public class Info {
2    public static String name =
3  "WazzUp";
4    public static int versionNr = 5;
5
6    public static String getName() {
7      return name;
8    }
9
10   public static int getVersionNr() {
11     return versionNr;
12   }
13 }
```

```
1  public class Friend {
2    public String name;
3
4    public Friend(String name) {
5      setName(name);  }
6
7    public String getName() {
8      return name;  }
9
10   public void setName(String
11 name) {
12     this.name = name;  }
13
14   public String toString() {
15     return getName();
16   }
17 }
```

Machen Sie sich zunächst klar, bei welchen Attributen es sich um Klassen- bzw. Objektattribute sowie Klassen- oder Objektmethoden handelt.

Geben Sie dann zu den folgenden Code-Abschnitten an, ob sie kompilieren. Wenn ein Code-Abschnitt nicht kompiliert erläutern Sie kurz, warum das der Fall ist. Wenn ein Code-Abschnitt kompiliert, dann geben Sie für jede Zeile an:

- Bei Zuweisungen: Was wird nach der Zuweisung in der Zielvariablen gespeichert?
- Bei Ausgaben: Was wird nach der Ausgabe auf dem Bildschirm ausgegeben?

Teil-aufgabe	Code-Abschnitt	Lösungsvorschlag
a)	`int v = Info.getVersionNr();` `System.out.println(v);`	`v = 5` `Ausgabe: 5`
b)	`Info.setName("NEW");`	`Geht nicht: Methode existiert nicht`
c)	`String f2Name = Friend.getName();` `System.out.println(f2Name);`	`Geht nicht: getName() ist keine Klassen-Methode`

d)	`Friend f1 = new Friend("James Gosling");`
	`System.out.println(f1.toString());`
	`System.out.println(new Friend("Andrew Tanenbaum").getName());`
		`f1 speichert ein Objekt von Friend`
		`Ausgabe: James Gosling`
		`Ausgabe: Andrew Tanenbaum`

e)	`Friend f3 = new Friend("Bill Gates");`
	`Friend f4 = new Friend("Linus Torvalds");`
	`f3 = f4;`
	
	`String f3Name = f3.getName();`
	`System.out.println(f3Name);`
		`f3 speichert ein Objekt von Friend`
		`f4 speichert ein Objekt von Friend`
		`f3 speichert das Objekt von Friend, das auch in f4 gespeichert ist`
		`f3Name = Linus Torvalds`
		`Ausgabe: Linus Torvalds`

f)	`Friend f5 = new Friend();` `System.out.println(f5.toString());`	`Geht nicht: die Klasse Friend hat keinen leeren Konstruktor`

Aufgabe 3 (Programmieren – Verwaltung von Sparkonten)

In der vorherigen Übung haben Sie bereits ein Programm zur Verwaltung von Konten implementiert. Die Bank hat nun neue Anforderungen, die Sie in ihrer Software umsetzen sollen. Neben einem normalen Konto bietet die Bank neuerdings auch Sparkonten an. Sparkonten unterscheiden sich von normalen Konten dadurch, dass das Guthaben auf Sparkonten jährlich verzinst wird. Die Bank bietet hierzu allen Kunden die gleichen Konditionen bezüglich des Zinssatzes an. Ändern Sie Ihr Programm aus Übung 6 wie folgt ab:

Hinweis: *Sie können den Programmcode der Musterlösung von Übung 6 in Moodle herunterladen.*

a) Erstellen Sie eine neue Klasse `Sparkonto`. Diese soll die gleiche Funktionalität aufweisen, wie normale Konten. Kopieren Sie also einfach den Programmcode der Klasse `Konto` und ändern Sie den Namen der Klasse.

b) Zu jedem `Sparkonto` soll nun der aktuell gültige Zinssatz gespeichert werden. Der Zinssatz kann sich jedoch von Jahr zu Jahr ändern. Fügen Sie der Klasse `Sparkonto` ein Attribut `zinssatz` vom Typ `double` hinzu. Überlegen Sie dabei, ob hier ein Objekt- oder ein Klassenattribut sinnvoller ist.

c) Implementieren Sie in der Klasse `Sparkonto` die Methode `verzinsen()`. Der Methode werden keine Parameter übergeben und sie gibt keinen Wert zurück. Die Methode soll das aktuell auf dem Sparkonto vorhandene Guthaben entsprechend dem `zinssatz` verzinsen, d.h. das Guthaben entsprechend erhöhen. Für die Zahlung der Zinsen auf das Konto (und damit die Erhöhung des Kontostands) soll die Methode `einzahlen(double betrag)` genutzt werden.

d) Implementieren Sie nun in der Klasse `Sparkonto` die folgende Klassenmethode: `jahresabschluss(Sparkonto[] konten).` Diese Methode soll für jedes im übergebenen Array `konten` enthaltene Sparkonto die Methode `verzinsen()` aufrufen.

e) Ändern Sie nun die Klasse `Kontoverwaltung` so ab, dass im Hauptprogramm (main-Methode) nun fünf `Sparkonto`-Objekte (und nicht mehr vom Typ `Konto`) erstellt werden. Deklarieren Sie nun ein Array, das `Sparkonto`-Objekte aufnimmt, und weisen alle fünf Sparkonto-Objekte diesem Array zu. Auf den Elementen des Arrays sollen nun die gleichen Ein- und Auszahlungen wie in der vorherigen Übung ausgeführt werden und danach alle Kontostände ausgegeben werden. Danach soll der Zinssatz für Sparkonten auf 0.04 gesetzt und ein Jahresabschluss durchgeführt werden, d.h. die Methode `jahresabschluss(Sparkonto[] konten)` soll mit dem erstellten Array von `Sparkonto`-Objekten aufgerufen werden. Geben Sie danach erneut alle Kontostände aus. Führen Sie danach noch einen weiteren Jahresabschluss mit einem Zinssatz von 0.1 aus und geben Sie wiederum die Kontostände aus.

Hinweis: *Die Ausgabe der Konten können Sie in eine statische Methode innerhalb der Klasse Kontoverwaltung auslagen, damit Sie nicht jedes Mal eine neue Schleife schreiben müssen.*

Lösung Aufgabe 3

```java
// Aufgabenteil a)
public class Sparkonto {
	int kontonummer;
	double kontostand;
	// Attribut für Aufgabenteil b)
	static double zinssatz;

	public Sparkonto(int nr) {
		kontonummer = nr;
		kontostand = 0.0;
	}

	public boolean einzahlen(double betrag) {
		if (betrag >= 0) {
			kontostand = kontostand + betrag;
			return true;
		} else {
			return false;
		}
	}

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

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

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

```java
public class Kontoverwaltung {

	// Aufgabenteil e)
	public static void main(String[] args) {
		Sparkonto erstesKonto = new Sparkonto(111);
		Sparkonto zweitesKonto = new Sparkonto(222);
		Sparkonto drittesKonto = new Sparkonto(333);
		Sparkonto viertesKonto = new Sparkonto(444);
		Sparkonto fuenftesKonto = new Sparkonto(555);

		Sparkonto[] konten = {erstesKonto, zweitesKonto,
		drittesKonto, viertesKonto, fuenftesKonto};

		konten[0].einzahlen(500.00);
		konten[1].einzahlen(-5.00);
		konten[2].einzahlen(46.50);
		konten[3].auszahlen(5.60);
		konten[0].auszahlen(42.00);
		konten[4].einzahlen(4.33);

		Kontoverwaltung.kontenAusgeben(konten);

		Sparkonto.zinssatz = 0.04;
		Sparkonto.jahresabschluss(konten);
		System.out.println("--- Jahresabschluss
		durchgeführt ---");
		Kontoverwaltung.kontenAusgeben(konten);

		Sparkonto.zinssatz = 0.1;
		Sparkonto.jahresabschluss(konten);
		System.out.println("--- Jahresabschluss
		durchgeführt ---");
		Kontoverwaltung.kontenAusgeben(konten);
	}

	public static void kontenAusgeben(Sparkonto[] konten) {
		for (int i = 0; i < konten.length; i++) {
			System.out.println("Kontostand für Kontonr. " +
			konten[i].kontonummer + ": " + konten[i].kontostand);
		}
	}
}
```

Kontostände nach der letzten Verzinsung:

```
Kontostand für Kontonr. 111: 523.952
Kontostand für Kontonr. 222: 0.0
Kontostand für Kontonr. 333: 53.196
Kontostand für Kontonr. 444: 0.0
Kontostand für Kontonr. 555: 4.953519999999999
```

Aufgabe 4 (Suchen von Zeichenketten)

a) Erstellen Sie für die Klasse `MyString` eine statische Methode `occurrences`, die zwei Parameter vom Typ `String` übergeben bekommt und zurückgibt, wie häufig die erste der übergebenen Zeichenketten in der zweiten vorkommt. Nutzen Sie dazu die unten angegebenen Objektmethoden der Klasse String.

Beispiel: *der folgende Aufruf soll den ganzzahligen Wert 1 ergeben:*
```
MyString.occurrences(„ei“, „JaVa isT eIn KafFeE, eiNe InSEl unD Eine ProgRamMiersprAche!“)
```

b) Schreiben Sie eine weitere statische Methode `caseInsensitiveOccurrences`, die das gleiche macht wie die Methode aus Teilaufgabe (a), aber dabei nicht zwischen Groß-/Kleinschreibung unterscheidet. Z.B. kommt dann „a“ drei Mal in der Zeichenkette „Auf nach DA!“ vor. Überlegen Sie auch, ob Sie die schon bestehende Methode aus Teilaufgabe (a) hierzu wiederverwenden können.

Beispiel: *der folgende Aufruf soll den ganzzahligen Wert 3 ergeben:*
```
MyString.caseInsensitiveOccurrences(„ei“, „JaVa isT eIn KafFeE, eiNe InSEl unD Eine ProgRamMiersprAche!“)
```

Nützliche Objektmethoden für Objekte der Klasse `String` (z.B. für `String s = „Haus“`):

- `s.indexOf(String searchValue, int startIndex);`
  searchValue: die zu suchende Zeichenkette
  startIndex (optional): der Index, an welchem die Suche beginnen soll
  Rückgabewert: Index, an dem die zu suchende Zeichenkette in `s` vorkommt. Wenn diese nicht vorkommt, wird -1 zurückgegeben.
  Bsp: `s.indexOf(„ei“, 0); // ergibt -1`
- `s.toLowerCase();`
  Rückgabewert: String, welcher `s` entspricht, jedoch nur Kleinbuchstaben enthält
  `s.toLowerCase(); // ergibt „haus“`
- `s.toUpperCase();`
  Rückgabewert: String, welcher `s` entspricht, jedoch nur Großbuchstaben enthält
  `s.toUpperCase(); // ergibt „HAUS“`

Lösung Aufgabe 4

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

		String x = "ei";
		String y = "JaVa isT eIn KafFeE, eiNe InSEl unD Eine
		ProgRamMiersprAche!";

		System.out.println(MyString.occurrences(x, y));
		System.out.println(MyString.caseInsensitiveOccurrences(x, y));
	}

	// Aufgabenteil a)
	public static int occurrences(String a, String b) {
		int counter = 0; // Ergebnisvariable
		int searchFromIndex = -1;
		do {
			searchFromIndex = b.indexOf(a, searchFromIndex + 1);
			if (searchFromIndex >= 0) {
				counter++;
			}
		} while (searchFromIndex >= 0);
		return counter;
	}

	// Aufgabenteil b)
	public static int caseInsensitiveOccurrences(String a, String b) {
		String lowerCaseA = a.toLowerCase();
		String lowerCaseB = b.toLowerCase();
		int n = MyString.occurrences(lowerCaseA, lowerCaseB);
		return n;
	}

}
```