Übung 9

TECHNISCHE UNIVERSITÄT DARMSTADT

Aufgabe 1 (Wissen)

a) Welche Sichtbarkeits-Modifier sind Ihnen bekannt und welche Zugriffe erlauben sie jeweils auf die Attribute und Methoden eines Objekts?

b) Ein Kollege präsentiert Ihnen seine Implementierung der Klasse `Student`, die im Rahmen einer Universitäts-Verwaltungssoftware verwendet werden soll. Leider hat Ihr Kollege den Zugriff auf Objektattribute nicht ganz richtig geregelt: Auf jedes Attribut kann sowohl über den Konstruktor als auch über Getter/Setter zugriffen werden. Das ist nicht in allen Fällen sinnvoll. Beantworten Sie für jedes Attribut die folgenden drei Fragen mit Begründung:
   a. Sollte das Attribut über den Konstruktor belegt werden?
   b. Sollte das Attribut über eine Getter-Methode lesbar gemacht werden?
   c. Sollte das Attribut über eine Setter-Methode beschreibbar gemacht werden?

```java
public class Student {
        private int matrikelNr;
        private String name;
        private String adresse;
        private String thesisTitel;

        public Student(int matrikelNr, String name, String adresse, String thesisTitel) {
                this.matrikelNr = matrikelNr;
                this.name = name;
                this.adresse = adresse;
                this.thesisTitel = thesisTitel;
        }

        public int getMatrikelNr() {
                return matrikelNr;    }

        public void setMatrikelNr(int matrikelNr) {
                this.matrikelNr = matrikelNr;       }

        public String getName() {
                return name;   }

        public void setName(String name) {
                this.name = name;     }

        public String getAdresse() {
                return adresse;       }

        public void setAdresse(String adresse) {
                this.adresse = adresse;      }

        public String getThesisTitel() {
                return thesisTitel;   }

        public void setThesisTitel(String thesisTitel) {
                this.thesisTitel = thesisTitel;     }
}
```

Aufgabe 2 (Sichtbarkeit)

Gegeben ist die folgende Klasse `Student`:

```
 1  package gdp.access.students;
 2
 3  public class Student {
 4        private int id = 1;
 5        private int kontonummer;
 6        private String adresse;
 7
 8        int getKontonummer() {
 9              return kontonummer;
10        }
11
12        void setKontonummer(int kontonr) {
13              kontonummer = kontonr;
14        }
15
16        public String getAdresse() {
17              return adresse;
18        }
19
20        public void setAdresse(String adr) {
21              adresse = adr;
22        }
23  }
```

Die nachfolgenden Klassen versuchen auf die Daten eines Studenten (genauer: eines Objekts der Klasse `Student`) zuzugreifen. Einige dieser Datenzugriffe werden durch sorgfältig gewählte Sichtbarkeitsstufen verhindert. Markieren Sie in der ersten Spalte [X] der nachfolgenden Klassen die Zeilen Code, die aufgrund von Sichtbarkeitseinstellungen nicht möglich sind!

a)

[X]		
	1	`package gdp.access.external;`
	2	
	3	`import gdp.access.students.Student;`
	4	
	5	`public class GezDurchDieStadtverwaltung {`
	6	` public void datenAbgreifen(Student s) {`
X	7	`  int id = s.id;`
X	8	`  int k = s.getKontonummer();`
	9	`  String a = s.getAdresse();`
	10	` }`
	11	`}`

b)

[X]		
	1	`package gdp.access.external;`
	2	
	3	`import gdp.access.students.Student;`
	4	
	5	`public class GezMitarbeiter extends Student {`
	6	` public void datenAbgreifen(Student s) {`
X	7	`  int id = s.id;`
X	8	`  int k = s.getKontonummer();`
	9	`  String a = s.getAdresse();`
	10	` }`
	11	`}`

c)

[X]		
	1	`package gdp.access.students;`
	2	
	3	`public class GezInformant extends Student {`
	4	` public void datenAbgreifen(Student s) {`
X	5	`  int id = s.id;`
	6	`  int k = s.getKontonummer();`
	7	`  String a = s.getAdresse();`
	8	` }`
	9	`}`

Aufgabe 3 (Gültigkeit von Variablen)

Bearbeiten Sie die untenstehenden Fragen in Bezug auf den folgenden Programmcode:

```
 1  public class Scope {
 2        public static int foo = 42;
 3
 4        public static void main(String[] args) {
 5              System.out.println(foo);
 6              int foo = 0;
 7              System.out.println(foo);
 8              foo = Scope.foo + 1;
 9              {
10                    int bar = 5;
11                    System.out.println(bar);
12                    System.out.println(foo);
13
14                    double foo = 0.0;
15              }
16              System.out.println(foo);
17              bar = foo + 1;
18              qux(Scope.foo);
19        }
20
21        static void qux(int baz) {
22              int foo;
23              {
24                    int baz;
25                    System.out.println(baz);
26              }
27        }
28  }
```

a) Finden Sie im obigen Programm die **drei Fehler**, die das Kompilieren des Quelltextes verhindern. Geben Sie in der folgenden Tabelle für jeden Fehler die Zeile und eine Beschreibung des Fehlers an.

Zeile	Fehler
17	Variable "bar" wird nicht definiert.
14	Auseinandersetzung des Definitions "foo".
24	Auseinandersetzung des Definitions "baz"

b) Tragen Sie in der folgenden Tabelle die Ausgaben des Programms unter Angabe der Zeilennummer ein.
Nehmen Sie für diese Teilaufgabe an, dass alle Zeilen, die einen Syntaxfehler enthalten, auskommentiert sind, sodass der Programmcode trotz der syntaktischen Fehler kompiliert werden kann.

Zeile	Ausgabe
5	42
7	0
11	5
12	43
16	0 X 42
25	42

Aufgabe 4 (Buchvergleich)

Gegeben sind die beiden folgenden Klassen:

```java
public class Book {
	private String title;

	public Book(String title) {
		this.title = title;
	}

	public String getTitle() {
		return title;
	}

	public void setTitle(String title) {
		this.title = title;
	}
}
```

```java
public class BookManagementSystem {

	public static void main(String[] args) {
		Book ir1 = new Book("Software Ecosystems");
		Book ir2 = new Book("Software Ecosystems");

		// Check if both books are equal.
		if (ir1 == ir2) {
			System.out.println("The books are duplicates!");
		} else {
			System.out.println("The books are NO duplicates!");
		}
	}
}
```

a) Warum führt der Vergleich der beiden Objekte `ir1` und `ir2` nicht zum gewünschten Ergebnis (die beiden Bücher sollten eigentlich als Duplikate erkannt werden)?

b) Implementieren Sie in der Klasse `BookManagementSystem` eine Klassenmethode `isDuplicate(Book b1, Book b2)`, die einen `boolean`-Rückgabewert besitzt und zwei Parameter vom Datentyp `Book` entgegennimmt. Diese Methode soll `true` zurückgeben, wenn die Titel der übergebenen Bücher gleich sind und `false`, wenn sie es nicht sind.

c) Implementieren Sie die `main`-Methode in der Klasse `BookManagementSystem` nun so, dass die von ihnen implementierte Methode `isDuplicate(Book,Book)` verwendet wird und die beiden Duplikate erkennt.

d) Sie stellen fest, dass eine statische Implementierung der Methode `isDuplicate(Book,Book)` in der Klasse `BookManagementSystem` keine besonders gute Lösung ist. Zum einen, weil die Klasse `BookManagementSystem` nicht als Sammelstelle für alle denkbaren Methoden gedacht ist. Zum anderen, weil die Klasse `Object` (von der alle anderen Klassen erben, also auch die Klasse `Book`) für die Überprüfung auf Gleichheit eine eigene Methode bereitstellt:

```
boolean equals(Object obj)
   Indicates whether some other object is "equal to" this one.
```

Implementieren Sie das Programm nun so, dass die Klasse `Book` die Objekt-Methode `equals(Object)` überschreibt. Ändern Sie die `main`-Methode der Klasse `BookManagementSystem` so ab, dass statt der statischen Methode `isDuplicate(Book,Book)` die `equals`-Methode der `Book`-Objekte verwendet wird, um dieselbe Funktionalität zu realisieren.

Aufgabe 1.

a)

- Public: Könnten von aller Klasse nd Pakete zugegriffen werden.
- Protected: Geschützte Variablen / Methoden. Sie dürfen nur von der eigenen Klasse, seine vererbten Klasse, von Klassen in eigenen Pakete zugegriffen werden.
- Package (Keine Sichtbarkeitsangabe): Könnten nur im eigenen Paket zugegriffen werden.
- Private: Könnten nur innerhalb der eigene Klasse zugegriffen werden.

b) In eigene Klasse könnten allen Attribute durch den Konstruktur belegt werden.

Aufgabe 4:

a) Statt "==" soll "equal of" verwendet werden. Anhand "==" werden nur Addresse der Variable miteinander verglichen.