PROGRAMMIER-
TECHNIK I
ÜBUNG

12

Die Besprechung der Aufgabenblätter 11 und 12 findet im Neuen Jahr am 13/14 Januar 2021 statt. Bitte versuchen Sie die Aufgaben ohne Rechner zu lösen. Das Übungsblatt enthält auch die Lernziele aus den jeweiligen Vorlesungen sowie die wichtigsten Konzepte und Begriffe.

Übung 0 – Lernziele

- K1 - Sie kennen ein paar **Teilgebiete des Software Engineerings**
  - Nennen Sie zwei Teilgebiete des Software Engineerings und erläutern Sie diese jeweils kurz (1-3 Sätze).
- K2 - Sie können erläutern was zum **Programmieren** dazu gehört
  - Erläutern Sie warum zum Programmieren mehr dazu gehört als „nur“ das Schreiben von Code. Geben Sie hierfür zwei Aspekte an, die im Zusammenhang mit dem Schreiben von Code berücksichtigt werden müssen und erläutern Sie diese kurz (jeweils 1-3 Sätze).
- K1 - Sie können erklären was ein **Compiler** ist
  - Erläutern Sie kurz was ein Compiler ist. Beantworten Sie dabei folgende Fragen:
    a) Welchen Input benötigt und welchen Output erzeugt ein Compiler? Berücksichtigen Sie bei Ihrer Antwort den Compileransatz und den hybriden Ansatz!
    b) Welche Komponenten hat ein Compiler typischerweise? Nennen Sie zwei Komponenten und erläutern Sie diese kurz (1-3 Sätze pro Komponente).
    c) Worin unterscheiden sich Compiler und Interpreter? Nennen Sie zwei Merkmale und erläutern Sie diese kurz (1-3 Sätze pro Merkmal).
- K2 - Sie können erklären was die **Java Virtual Machine** und **Bytecode** sind
  - Welche Aufgabe hat die Java Virtual Machine? Erläutern Sie diese kurz (1-3 Sätze).
  - Worin unterscheiden sich die Java Virtual Machine und der Java Compiler? Nennen Sie zwei Aspekte und erläutern Sie diese kurz (1-3 Sätze pro Aspekt).
- K2 - Sie können erläutern wie aus **Java-Code** ein **ausführbares Programm** entsteht
  - Erläutern Sie die Schritte, die notwendig sind um von Java-Quellcode zum ausführbaren Programm zu kommen. Erläutern Sie in diesem Zusammenhang das Konzept der Plattformunabhängigkeit.
- K2 - Sie können erklären warum Java plattformunabhängig ist
  - Erläutern Sie warum Java eine plattformunabhängige Sprache ist?
  - Was versteht man unter dem Compiler- und unter dem Interpreteransatz? Erläutern Sie beide Konzepte kurz (1-3 Sätze jeweils)!

- Welche Kriterien spielen bei der Auswahl einer Programmiersprache für ein bestimmtes Problem oder für eine bestimmte Domäne eine große Rolle? Nennen Sie zwei Kriterien und erläutern Sie diese jeweils kurz (1-3 Sätze pro Aspekt).

– K1 - Sie wissen was eine **IDE** und **eclipse** sind

- Was versteht man unter einer IDE (1-3 Sätze)? Geben Sie ein Beispiel für eine IDE für Java.

– K3 - Sie haben Ihr erstes Java-Programm geschrieben und „**zum Laufen gebracht**“

Begriffe/Konzepte

Compiler, Interpreter, Compileransatz, Interpreteransatz, hybrider Ansatz, Software Engineering, Programmieren, IDE, Plattformunabhängigkeit, Offenheit (bei Programmiersprachen), Objektorientierung, JRE, Java-Compiler, Bytecode, JVM Java Virtual Machine

PROGRAMMIER-
TECHNIK I
ÜBUNG

12

VORLESUNG 4 – Kontrollstrukturen

- K3 - Sie können einfache Algorithmen/Programme zu einer gegebenen Aufgabenstellung mit Hilfe von Kontrollstrukturen formulieren

Begriffe/Konzepte

Casting (implizit/explizit), KISS, Sequenz, Blöcke, Gültigkeitsbereich, if-Verzweigung/Entscheidungsanweisung, dangling if, switch/case/default, Schleifen, Iterationen, while, do-while, for-Schleife, Methodenaufruf

Übung 4.1. – Programmverständnis, Schleifen

Gegeben sei folgendes Java-Programm.

```
01 public class MachWas {
02
03     public static int machWas (int zahl) {
04           int rest, min, ziffer;
05           int anzahlDurchläufe = 0;
06           rest = zahl;
07
08           if (zahl == 0) {
09                 min = 0;
10           } else {
11                 min = 9;
12           }
13
14           while (rest > 0) {
15                 ziffer = rest % 10;
16                 rest = rest / 10;
17                 if (ziffer < min) {
18                       min = ziffer;
19                 }
20           }
21           return min;
22     }
23
24     public static void main(String[] args) {
25           System.out.println(machWas (15));
26           System.out.println(machWas (30985));
27           System.out.println(machWas (0));
28     }
29 }
```

a) Was ist jeweils die Ausgabe in den Zeilen 25 – 27?

Zeile	Ausgabe
25	
26	
27	

b) Beschreiben Sie natürlichsprachlich was die Methode „machWas“ macht.
c) Was passiert, wenn Sie die Methode mit negativen Zahlen aufrufen? Korrigieren Sie die Methode derart, dass das Ergebnis auch für negative Zahlen korrekt ist.
d) Wie oft wird die Schleife ab Zeile 14 durchlaufen? Wovon hängt die Anzahl der Durchläufe ab?
e) Geben Sie ein Beispiel für welches die Schleife genau 4-mal durchlaufen wird.

VORLESUNG 6 - Arrays

– K3 - Sie verstehen den Unterschied zwischen einer elementaren Variablen und einer Referenzvariablen (auch Übung 6.1)

- Erläutern Sie den Unterschied zwischen elementaren Datentypen und Referenzdatentypen. Gehen Sie dabei auf die Konzepte „Stack“ und „Heap“ ein!

– K3 - Sie verstehen was ein Array ist und Sie können ein Array anlegen, durchlaufen und kopieren (Übung 6.1)

- Was sind Arrays?
- Können in einem Array Variablen unterschiedlichen Typs gespeichert werden?
- Können in einem Array Referenzvariablen desselben Typs gespeichert werden?
- Kann die Länge eines Arrays dynamisch (d.h. während der Laufzeit) geändert werden?

– K3 - Sie verstehen den Unterschied zwischen einer echten Kopie und einer Referenzkopie (Übung 6.1)

Begriffe/Konzepte

Referenzdatentypen, Stack, Heap, mehrdimensionale Arrays, echte Kopie, Referenz

Übung 6.1. – Arrays - Arrays traversieren, Kopie und Referenz

Gegeben sei folgendes Java-Programm.

```
01 public class Arrays {
02
03    public static void main(String[] args) {
04          int [] myArr1 = {1, 2, 3, 4, 5};
05          int [] myArr2;
06          int sum = 0;
07
08          myArr2 = myArr1;
09
10          for (int i = 0; i<myArr2.length; i++) {
11                myArr2 [i] = myArr1[i] - 1;
12          }
13
14          for (int i = 0; i<myArr1.length; i++) {
15                sum = sum + myArr1[i];
16          }
17          System.out.println (sum);
18
19          for (int i = 0; i<myArr2.length; i++) {
20                sum = sum + myArr2[i];
21          }
22
23          System.out.println (sum);
24          // Ab hier erweitern für Teilaufgabe 4
25   }
26}
```

a) Was ist die Ausgabe in Zeile 17?

b) Was ist die Ausgabe in Zeile 23?

c) Erläutern Sie den Unterschied zwischen Referenzdatentypen und primitiven Datentypen! Welche der Variablen im obigen Programm sind von einem elementaren Datentyp und welche von einem Referenzdatentyp?

d) Erweitern Sie das obige Programm ab Zeile 24. Erzeugen Sie eine echte Kopie von **`myArr1`**!

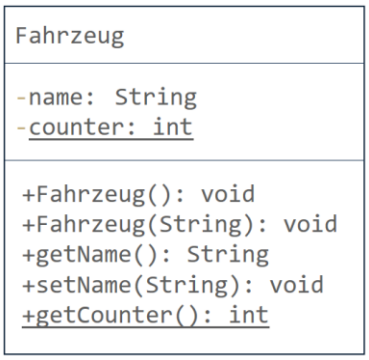
VORLESUNG 7 – KOMMENTARE, KLASSEN – Übung 7.1

- K3 - Sie können **Klassen** erstellen und auf diese zugreifen
- K3 - Sie können an einem Beispiel erklären, warum **Datenkapselung/Information Hiding** ein grundlegendes Prinzip ist
- K3 - Sie kennen den Unterschied zwischen Instanz(Methoden/Variablen) und Klassen(Methoden/Variablen)
- K3 - Sie können **Konstruktoren** erstellen

- **Begriffe/Konzepte**
- Referenzdatentypen, Stack, Heap, Referenz, Klasse, Instanz, Instanzvariablen, Instanzmethoden, Klassenvariablen, Klassenmethoden, UML, Klassendiagramm, Konstruktor, Information/Data Hiding, Konstruktoren überladen

Übung 7.1. – Klassen – Klassen aus einer UML-Spezifikation implementieren, Instanz- und Klassenvariablen, Information Hiding, einfache Konstruktoren

Gegeben sei folgende Klassenspezifikation in UML-Darstellung.



a) Was sind Attribute? Was sind Methoden? Erläutern Sie beide Begriffe am obigen Beispiel!
b) Erläutern Sie am obigen Beispiel den Unterschied zwischen Klassen- und Instanzvariablen sowie zwischen Klasse- und Instanzmethoden?
c) Erläutern Sie am obigen Beispiel was ein Konstruktor ist! Implementieren Sie die obige Klasse wie abgebildet. Erstellen Sie zwei Konstruktoren, einen parameterlosen Konstruktor und einen Konstruktor, der den Namen des Fahrzeugs initialisiert.
d) Erläutern Sie an diesem Beispiel was es bedeutet, Konstruktoren zu überladen und warum dies sinnvoll ist.
e) Implementieren Sie die Methoden
    a. `setName`, die den Namen eines Fahrzeugs setzt
    b. `getName`, die den Namen des Fahrzeugs zurückgibt.
f) Implementieren Sie eine Methode `getCounter`, die die aktuelle Anzahl der Fahrzeuge zurückgibt.

g) Warum sollte man keine `setCounter`-Methode bereitstellen, die die aktuelle Anzahl der Fahrzeuge setzt?

h) Gegeben sei das nachfolgende Programm, welches die obige Klasse verwendet. Was passiert in Zeile 05? Erläutern Sie den Vorgang kurz und gehen Sie auf die Konstruktoren ein!

i) Ergänzen Sie die jeweiligen Ausgaben in den Zeilen

Zeile	Ausgabe
11	
12	
13	

```
01 public class FahrzeugeTestlauf {
02
03   public static void main(String[] args) {
04
05        Fahrzeug pkw = new Fahrzeug("PKW");
06        Fahrzeug motorrad;
07
08        motorrad = pkw;
09        motorrad.setName("Motorrad");
10
11        System.out.println (motorrad.getName());
12        System.out.println (pkw.getName());
13        System.out.println (Fahrzeug.getCounter());
14   }
15 }
```

j) Wie viele Instanzen der Klasse „Fahrzeug“ leben zur Laufzeit zum Zeitpunkt der Ausführung der Zeile 10?

k) Erläutern Sie den Unterschied zwischen Klassen und Instanzen am obigen Beispiel!

l) Was ist Information Hiding? Erläutern Sie das Konzept am obigen Beispiel!

VORLESUNG 8 – Objektorientierte Konzepte

- Sie können **Konstruktoren** erstellen (Übung 8.1)
- Sie können die Prinzipien der **Vererbung** und **Generalisierung** anwenden um ein Beispiel aus der realen Welt zu modellieren (Übung 8.2)
    - Nennen Sie zwei Vorteile der Objektorientierung und erläutern diese an einem Beispiel!
- Sie können den Unterschied zwischen **Überladen** und **Überschreiben** erläutern (Übung 8.2)
- Sie wissen welche **Zugriffsrechte** vergeben werden können und können diese sinnvoll einsetzen (Übung 8.1, 8.2)

Begriffe/Konzepte

Vererbung, Generalisierung, Überladen, Überschreiben, Modifier, Zugriffsrechte, public, protected, private, package (default), Oberklasse, Unterklasse, Klassenhierarchie,

Übung 8.1. – Klassen – Konstruktoren

Gegeben sei folgendes Java-Programm, welches die Klasse „Buch“ implementiert und das Java-Programm „BuchTestlauf“ welches die Klasse Buch verwendet.

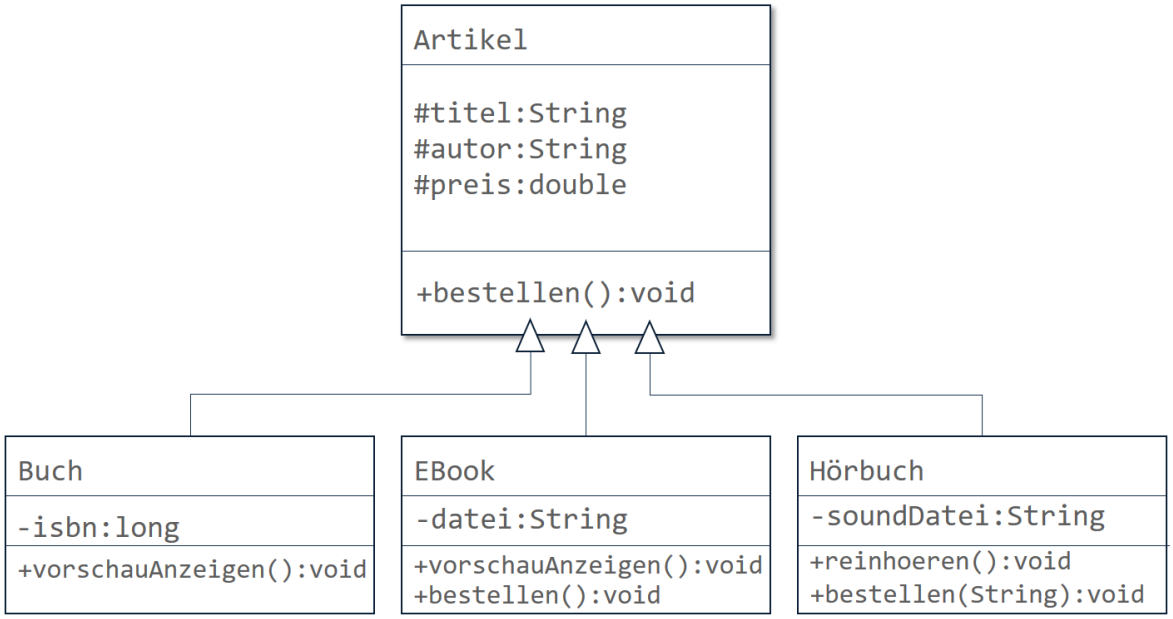
```
public class Buch {
	private String titel = "";
	private double preis = 0.0;

	public void Buch () {
		this.titel = "Default Titel";
		this.preis = 44.99; }

	public String getTitel() {
		return titel; }

	public void setTitel(String titel) {
		this.titel = titel; }

	public double getPreis() {
		return preis; }

	public void setPreis(double preis) {
		this.preis = preis; }
}
```

```
01 public class BuchTestlauf {
02
03    public static void main(String[] args) {
04          Buch meinBuch = new Buch ();
05          System.out.println (meinBuch.getTitel());
06          System.out.println (meinBuch.getPreis());
07    }
08
09 }
```

a) Was ist jeweils die Ausgabe in den Zeilen 05 und 06 und warum?
b) Erläutern Sie das Konzept des Default-Konstruktors!
c) Schreiben Sie einen Konstruktor, der alle Werte mit den übergebenen Parametern initialisiert!
d) Erläutern Sie die Schlüsselworte `private` und `public` an Hand des obigen Beispiels!

Übung 8.2. – Objektorientierte Konzepte, Vererbung, Generalisierung, Überladen

Gegeben sei folgende Spezifikation in Form eines UML-Diagramms.



a) Erläutern Sie das Konzept der Vererbung am obigen Beispiel! Verwenden Sie hierbei die Begriffe Ober- und Unterklasse.

b) Welche Methoden werden jeweils in den Klassen Buch, EBook, Hörbuch geerbt, überschrieben oder überladen? Welche Methoden kommen in der jeweiligen Unterklasse hinzu?

	Buch	EBook	Hörbuch
Geerbte Methoden			
Hinzugekommene Methoden			
Überschriebenen Methoden			
Überladene Methoden			

c) Gegeben sei das folgende Java-Programm, welches ein Teil der obigen Spezifikation implementiert. Welche der Zeilen 27 und 28 kann kompiliert werden? Warum?

```
01 public class Artikel {
02    protected String titel;
03    protected String autor;
04    protected double preis;
05
06    // weiterer wichtiger Code
07
08    public void bestellen () {
09          System.out.println ("Artikel wird wie gewünscht bestellt");
10    }
11 }

12 public class Hoerbuch extends Artikel {
13    private String soundDatei;
14
15    // weiterer wichtiger Code ist hier ausgeblendet
16
17    public void reinhören () {
18          // macht was
19    }
20
21    public void bestellen (String text) {
22          System.out.println ("Wir bearbeiten gerade Ihre Anmerkung. " + text);
23    }
24 }

24 public class ArtikelTestlauf {
25    public static void main(String[] args) {
26          Hoerbuch meinHoerbuch = new Hoerbuch ();
27          meinHoerbuch.bestellen();
28          meinHoerbuch.bestellen("Notiz an Verkäufer.");
29    }
30 }
```

d) Erweitern Sie die Klasse `Hoerbuch` um einen Konstruktor, der alle Instanzvariablen (eigene und geerbt) initialisiert.

e) Warum ist der Aufruf in Zeile 26 nach Erweiterung des Konstruktors in Aufgabe d) nicht mehr valide? Welche Möglichkeiten haben Sie den Fehler zu korrigieren?

f) Erläutern Sie das Schlüsselwort `protected` am obigen Beispiel!