PROGRAMMIER-
TECHNIK I
≥ ÜBUNG ≤

Wiederholung TEIL III

VORLESUNG 2 – Digitaler Wandel, Informatik und Teilgebieten

- K1 - Sie können jeweils Stärken und Schwächen **von Mensch und Maschine** benennen.
  - Erläutern Sie an zwei Beispielen jeweils die Stärken von Mensch und Maschine (1-2 Sätze pro Beispiel).
  - Was versteht man unter dem Begriff Mensch-Maschine Kollaboration? (1-2 Sätze)
- K2 - Sie können an einem Beispiel erläutern wie durch **Vernetzung** immer größere Systeme entstehen
  - Erläutern Sie an einem Beispiel wie durch Vernetzung ein digitales Ökosystem entsteht? (3 – 6 Sätze).
- K1 - Sie kennen die Kerngebiete der Informatik und können Beispiele dafür nennen
  - Nennen Sie zwei Kerngebiete der Informatik und geben Sie jeweils ein Beispiel für dessen Inhalt.
- K2 - Sie können Vor- und Nachteile unterschiedlicher Compiler/Interpreter-Ansätze erläutern
  - Was verstehen Sie unter dem „hybriden Ansatz“ in Zusammenhang mit Compiler/Interpreter-Ansätzen? (2-3 Sätze)
  - Vergleichen Sie Compiler und Interpreteransatz. Gehen Sie dabei auf zwei Aspekte ein. (1-2 Sätze pro Aspekt).
- K1 Sie wissen, wo ein Java-Programm anfängt
- K2 - Sie können erklären was ein Datentyp ist (Siehe Wiederholungsblatt 2)
- K2 - Sie können erklären was primitive Datentypen sind (Siehe Wiederholungsblatt 2)

Begriffe/Konzepte

Mensch/Maschine (Stärken, Schwächen), angewandte Informatik, praktische Informatik, theoretische Informatik, technische Informatik, Compileransatz, Interpreteransatz, hybrider Ansatz, Compiler vs. Interpreter,

VORLESUNG 5 – Rekursion

- K2 - Sie können an einem Beispiel erläutern, was „**pass-by-copy**“ bedeutet.
  - Übung 5.1, 5.2
- K2 - Sie können erläutern, was das **Überladen** von Methoden bedeutet.
  - Erläutern Sie den Unterschied zwischen Überladen und Überschreiben von Methoden? Welches Ziel hat das Überladen? Welches Ziel hat das Überschreiben? (2-4 Sätze)
- K2 - Sie können erläutern, wie die Speicherverwaltung auf dem **Stack** funktioniert.
  - Nach welchem Prinzip werden Stack-Frames abgelegt und abgearbeitet (1-2 Sätze)? Welche Art von Variablen werden auf dem Stack gespeichert?
- K3 - Sie können Aufgabenstellungen/ Algorithmen rekursiv und iterativ formulieren
  - Übung 5.1
- K2 - Sie können die beiden Konzepte Rekursion vs. Iteration vergleichen
  - Vergleichen Sie die beiden Konzepte „Rekursion“ und „Iteration“ anhand von zwei Aspekten (1-2 Sätze pro Aspekt).

Begriffe/Konzepte

Heap, Stack, Rekursion, pass-by-copy, Überladen von Methoden.

Übung 5. 1. - Rekursion

Gegeben sei das folgende Java-Programm.

```
01 public static int rekursiveMethode1 (int zahl) {
02     if (zahl < 0) {
03           throw new IllegalArgumentException ("Der Wert muss > 0 sein");
04     }
05
06     if (zahl < 10) {
07           return zahl;
08     }
09
10     int letzteZiffer = zahl % 10;
11     int restZahl = zahl / 10;
12
13     return rekursiveMethode1 (restZahl) + letzteZiffer;
14 }
```

a) Beschreiben Sie natürlichsprachlich was die Methode „rekursive Methode“ macht.
b) Was passiert in den Zeilen 06 – 08?
c) Welche Ausgabe erzeugt das Programm, wenn es mit den folgenden Parametern aufgerufen wird?

Eingabe	Ausgabe
`System.out.println (rekursiveMethode1(1234));`	
`System.out.println (rekursiveMethode1(1));`	
`System.out.println (rekursiveMethode1(0));`	
`System.out.println (rekursiveMethode1(-5));`	

Übung 5. 2. - Rekursion

Schreiben Sie eine Methode, welche rekursiv die Anzahl der Ziffern einer positiven natürlichen Zahl ermittelt. Ist der Eingabewert < 0 soll die `IllegalArgumentException` geworfen werden.

```
public static int anzahlZiffern (int zahl) { ...
```

Übung 5.3 – pass-by-copy

Gegeben Sie das folgende Java-Programm. Was ist die Ausgabe in Zeile 3 und in Zeile 10? Erläutern Sie das Konzept pass-by-copy an diesem Beispiel

```
1  public static void machWas (int x) {
2      x = x +1;
3      System.out.println ("Das x in machWas:" + x);
4  } // end machWas
5
6  public static void main(String[] args) {
7
8      int x = 10;
9      machWas (x);
10     System.out.println ("Das x nach machWas:" + x);
11
12 } // end main
```

VORLESUNG 9 – Polymorphie

- K3 - Sie können Konstruktoren schreiben und verstehen wie der Konstruktionsprozess entlang der Vererbungshierarchie stattfindet.

➔ Übung 9.1

- K2 - Sie kennen den Unterschied zwischen Konstruktoren und Methoden

➔ Worin unterscheiden sich Konstruktoren und Methoden? (1-2 Sätze)

- Sie können polymorphe Methoden designen und implementieren und können den Vorteil erläutern

➔ Übung 9.2

- Sie wissen, wie sich Methoden entlang der Vererbungshierarchie verhalten, welche Methoden aufgerufen werden, wenn in der Vererbungshierarchie Methoden sich überschreiben

➔ Übung 9.2

Begriffe/Konzepte

Polymorphie.

Übung 9. 1. – Konstruktoren im Vererbungsbaum

Gegeben seien die nachfolgenden Java-Klassen. Was ist die Ausgabe nach Ausführung der main-Methode im Programm ABCTestlauf?

```
public class Why {
	public Why () {
		System.out.println("Tell me why...");
	}
}

public class WhyNot {
	public WhyNot () {
		System.out.println("Why not?");
	}
}

public class A {
	public A () {
		System.out.println("Tell A");
	}
}

public class B extends A {
	public B () {
		System.out.println("Tell B");
	}
}


public class C extends B {

	Why why = new Why();
	WhyNot whyNot = new WhyNot();

	public C () {
		System.out.println("Tell C");
	}
}



public class ABCTestlauf {

	public static void main(String[] args) {
		C myLittleC = new C ();
	}
}
```

Übung 9. 2. – Polymorphie

Gegeben seien die nachfolgenden Java-Klassen.

```
public class Kurs {
	private String name;
	private String dozent;
	private boolean sale = false;

	public Kurs(String name, String dozent, boolean sale) {
		this.name = name;
		this.dozent = dozent;
		this.sale = sale;
	}

	public String getName() {
		return name;
	}

	public boolean isSale() {
		return sale;
	}

	public void buchen () {
		// Code zum Kurs buchen
		System.out.println ("Sie haben Ihren Kurs " + this.name + " " +
		"erfolgreich gebucht.");
	}

	public void infosZeigen () {
		// Code zum Kurs buchen
		System.out.println ("Kurs: " + this.name + ", " + this.dozent);
	}
}

public class Praesenz extends Kurs {

	public Praesenz(String name, String dozent, boolean sale) {
		super(name, dozent, sale);
	}
}

public class OnlineKurs extends Kurs {
	private String onlineLink = "Standard online Link auf die Homepage";

	public OnlineKurs(String name, String dozent, boolean sale) {
		super(name, dozent, sale);
	}

	public void infosZeigen () {
		// Code zum Kurs buchen
		super.infosZeigen();
		System.out.print("Weitere Infos finden Sie hier: ");
		System.out.println(onlineLink);
	}
}
```

```
public class OnlineSynchron extends OnlineKurs {
	private String konferenzLink = "Konferenz-Link";

	public OnlineSynchron(String name, String dozent, boolean sale) {
		super(name, dozent, sale);
	}
	public void infosZeigen () {
		// Code zum Kurs buchen
		super.infosZeigen();
		System.out.print("Wir freuen uns auf Sie in der Konferenz: ");
		System.out.println(konferenzLink);
	}
}

public class OnlineAsynchron extends OnlineKurs {

	public OnlineAsynchron(String name, String dozent, boolean sale) {
		super(name, dozent, sale);
	}

	public void buchen () {
		// Code zum Buchen
		System.out.println("Sie haben Ihren Kurs "+ this.getName()
		+ " erfolgreich gebucht."
				+  "Sie können gleich loslegen!");
	}
}
```

Erläutern Sie die Klassen graphisch. Welche Methoden werden in welcher Klasse überschrieben?

Gegeben sei weiterhin das nachfolgende Testprogramm:

```
public class MeineKurseTestlauf {

	public static void main(String[] args) {
		OnlineSynchron meinOnlineK1 =
			new OnlineSynchron("Englisch für Anfänger 1", "Dozent
			Englisch", false);
		OnlineSynchron meinOnlineK2 =
			new OnlineSynchron("Englisch für Anfänger 2", "Dozent
			Englisch", true);
		Praesenz meinPraesenzK =
			new Praesenz ("Englisch für Fortgeschrittene", "Dozent
			Englisch", true);
		OnlineAsynchron meinOnlineK3 =
			new OnlineAsynchron("Englisch Intermediate", "Dozent Englisch",
			false);

		Kurs [] warenkorb = new Kurs[4]; // Stelle 1
		warenkorb[0] = meinOnlineK1;
		warenkorb[1] = meinOnlineK2;
		warenkorb[2] = meinPraesenzK;
		warenkorb[3] = meinOnlineK3;

		for (int i = 0; i < warenkorb.length; i++) {
			warenkorb[i].buchen();
		}

		for (int i = 0; i < warenkorb.length; i++) {
			warenkorb[i].infosZeigen();
		}
	}
}
```

Was passiert an der markierten Stelle 1?

```
Kurs [] warenkorb = new Kurs[4];
```

Warum sind die nachfolgenden Zuweisungen valide?

```
		warenkorb[0] = meinOnlineK1;
		warenkorb[1] = meinOnlineK2;
		warenkorb[2] = meinPraesenzK;
		warenkorb[3] = meinOnlineK3;
```

Was ist die Ausgabe nach der ersten for-Schleife? Warum?

```
		for (int i = 0; i < warenkorb.length; i++) {
			warenkorb[i].buchen();
		}
```

Was ist die Ausgabe nach der zweiten for-Schleife? Warum?

```
		for (int i = 0; i < warenkorb.length; i++) {
			warenkorb[i].infosZeigen();
		}
```

Erweitern Sie das Testprogramm um eine Methode anzahlSale, die zu einem gegebenen Kurs-Array, die Anzahl der Kurse zurückliefert, die „im Sale“ sind?

VORLESUNG 10 – Lebensdauer, Gültigkeit

- K2 - Sie können an einem Beispiel den Unterschied zwischen Lebensdauer und Gültigkeit einer Variablen erläutern.

  - Was verstehen Sie unter der Gültigkeit einer Variablen? Kann der Gültigkeitsbereich statisch (d.h. zur Compilezeit) ermittelt werden?
  - Was verstehen Sie unter der Lebensdauer einer Variablen? Wie lange ist die Lebensdauer einer Instanzvariablen?
  - Wie lang ist die Lebensdauer eines Objekts? Wo werden Objekte gespeichert?
  - Was ist der Garbage Collector?

Begriffe/Konzepte

Lebensdauer, Gültigkeit, Gültigkeitsbereich, Sichtbarkeitsbereich, lokale Variable, Referenzvariable, Garbage Collection.

VORLESUNG 11 – Assoziationen

- K3 - Sie können Assoziationen erkennen, modellieren und in ein Java Programm umsetzen.

➔ Übung 9.2 erweitert

Begriffe/Konzepte

Assoziation, Assoziation vs. Vererbung.

Übung 11. 2. – Copy Konstruktor

a) Schreiben Sie einen Copy-Konstruktor für die Klasse „Kurs“ aus der Übung 9.2.
b) Die Klasse „Praesenz“ soll eine private Variable vom Typ „Raum“ speichern. In welcher Art von Beziehung stehen die Klassen „Raum“ und „Praesenz“?
c) Schreiben Sie einen Konstruktor für die Klasse Praesenz, der alle Parameter der Klasse initialisiert.
d) Schreiben Sie einen Copy-Konstruktor für die Klasse „Raum“ und für die Klasse Praesenz.