# Übung 5, Lösungsvorschlag

TECHNISCHE UNIVERSITÄT DARMSTADT

## Aufgabe 1 (Wissensfrage)

a) Was ist der Unterschied zwischen Klassen und Objekten?

b) Was sind Konstruktoren, welchen Zweck erfüllen sie und welche Eigenschaften haben alle Konstruktoren gemein? Wie werden Konstruktoren aufgerufen?

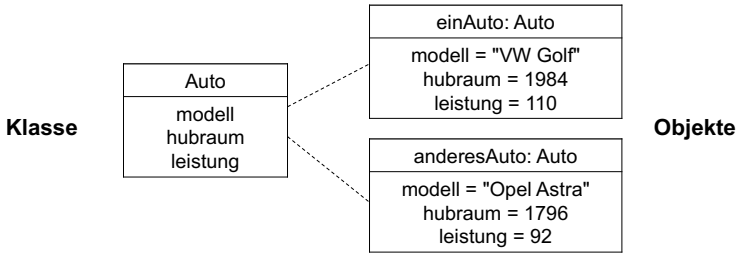
c) Aus welchen Bestandteilen setzt sich eine Klasse in Java zusammen?

d) Konstruktoren sind letztlich besondere Methoden. Sie haben gelernt, dass Methoden aus fünf Elementen bestehen. Welches wird bei Konstruktoren **nicht** mit angegeben?

## Lösung Aufgabe 1

a) **Klassen vs. Objekte**

- Die **Klasse** ist der **Datentyp**, die **Objekte** sind die **Werte**!
- Jedes Objekt ist **Instanz** genau einer Klasse, aber eine Klasse kann beliebig viele Instanzen besitzen
- Alle Objekte einer Klasse besitzen die gleichen **Methoden** und haben daher das gleiche Verhalten. Alle Objekte einer Klasse haben die gleichen **Attribute**, allerdings mit unterschiedlichen Werten (Zustand)




b) Konstruktoren sind spezielle Methoden, die bei der Erzeugung von Objekten aufgerufen werden.
Sie können (wie andere Methoden auch) Parameter übergeben bekommen und dienen der Erzeugung von Objekten und Initialisierung der Attribute mit Werten. Diese Werte können bspw. durch die Parameter übergeben werden.
Somit bereitet der Konstruktor den ersten Zustand vor, den jedes Objekt einer Klasse nach der Erzeugung haben soll.
Konstruktoren sind immer öffentlich, haben keinen Rückgabetyp und heißen wie ihre Klasse. Sie werden mittels des Schlüsselworts „new“ aufgerufen.

c) **Modifier** (public)
**Schlüsselwort** (class)
**Bezeichner** (Name der Klasse)
**Attribute** (Variablen verschiedener Datentypen, die zu einem Objekt gehören): Attribute sind das, was Objekte dieser Klasse „haben“.
**Methoden** (Funktionalität, die den Zustand eines jeden Objekts ändern kann oder Eigenschaften des Objekts liefert): Methoden sind das, was Objekte dieser Klasse „können“.

d) Bei Konstruktoren wird kein Rückgabetyp angegeben.

## Aufgabe 2 (Datum)

a) Schreiben Sie eine Klasse mit dem Bezeichner `Datum`. Jedes Objekt dieser Klasse soll über die Attribute `tag`, `monat` und `jahr` vom Datentyp `int` verfügen. Legen Sie hierzu die nötigen Attribute an. (Nutzen Sie zur Deklaration der Attribute `public int` …).

b) Erstellen Sie nun einen Konstruktor, der drei Parameter entgegennimmt. Die Datentypen der drei Parameter entsprechen den Datentypen der drei Attribute. Wählen Sie für die Parameter eigene Bezeichner, welche sich von den Bezeichnern der Attribute unterscheiden. Welchen Bezeichner muss der Konstruktor haben?

c) Im Konstruktor soll zunächst eine einfache Überprüfung vorgenommen werden, ob der Parameter für Monatszahl zwischen 1 und 12 liegt und die Tageszahl zwischen 1 und 31. Ist dies der Fall, so sollen die Attribute entsprechend gesetzt werden. Ansonsten wird eine Fehlermeldung auf der Konsole ausgegeben.

d) Erweitern Sie die Klasse `Datum` um eine Methode `druckeDatum`, welche das Datum auf der Konsole ausgibt. Nutzen Sie den Modifier `public`. Gibt die Methode einen Wert zurück? Welcher Rückgabetyp muss somit verwendet werden?

e) Schreiben Sie ein Hauptprogramm (`main`-Methode) in einer neuen, separaten Klasse mit dem Bezeichner `DatumApp`. In dieser Klasse benötigen Sie keine Attribute und keinen Konstruktor, aber eine main-Methode. Die main-Methode soll die folgenden Aufrufe enthalten:

   a. Deklarieren Sie eine Variable vom Typ `Datum.` Wählen Sie als Bezeichner `meinDatum`.

   b. Erzeugen Sie nun ein neues Objekt der Klasse `Datum` und weisen dieses der Variablen `meinDatum` zu. Rufen Sie hierzu den Konstruktor auf, der die Attribute des neu erzeugten Objekts auf das Datum des 1. Januar 1900 setzt.

   c. Lassen Sie sich das im Objekt gespeicherte Datum des soeben erzeugten Datum-Objekts mithilfe der eines Aufrufs der Methode `druckeDatum` ausgeben.

## Lösung Aufgabe 2

```
1    public class Datum {
2          // Attribute
3          public int jahr;
4          public int monat;
5          public int tag;
6
7          // Der Bezeichner des Konstruktors muss dem Klassennamen entsprechen
8          public Datum(int j, int m, int t) {
9                if (m < 1 || m > 12) {
10                     System.out.println("Der Monat muss eine Zahl zwischen 1
                 und 12 sein!");
11               } else if (t < 1 || t > 31) {
12                     System.out.println("Der Tag muss eine Zahl zwischen 1 und
                       31 sein!");
13               } else {
14                     // Parameterwerte werden den Attributen zugewiesen.
15                     jahr = j;
16                     monat = m;
17                     tag = t;
18               }
19         }
20
21         // Die Methode gibt zwar das Datum auf der Konsole aus,
22         // hat jedoch keinen Rückgabewert an den Aufrufer.
23         // Der Rückgabetyp ist damit void.
24         public void druckeDatum() {
25               System.out.println(tag + "." + monat + "." + jahr);
26         }
27   }
```

```
1     public class DatumApp {
2
3           public static void main(String[] args) {
4                 // Deklaration einer Variablen von Datentyp Datum
5                 Datum meinDatum;
6
7                 // Erzeugen eines neuen Datum-Objekts und Zuweisung der
                  Variablen meinDatum.
8                 meinDatum = new Datum(1900, 1, 1);
9
10                meinDatum.druckeDatum();
11          }
12    }
```

## Aufgabe 3 (Kontoverwaltung)

Sie sollen eine neue Kontoverwaltungssoftware für eine Bank implementieren. Sie haben sich überlegt, dass folgendes Vorgehen dabei sinnvoll wäre:

a) Implementieren Sie zunächst eine Klasse `Konto`. Ein Konto soll über ein Attribut `kontonummer` vom Datentyp `int` und ein Attribut `kontostand` vom Datentyp double verfügen. Verwenden Sie für beide Attribute den Modifier `public`.

b) Schreiben Sie nun einen Konstruktor für die Klasse `Konto`. Dem Konstruktor soll als einziger Parameter ein ganzzahliger Wert übergeben werden können, der dem Attribut `kontonummer` zugewiesen wird. Das Attribut `kontostand` soll beim Anlegen eines neuen Kontos immer auf 0 gesetzt werden.

c) Auf die `Konten` soll natürlich auch Geld eingezahlt werden können. Schreiben Sie dazu in der Klasse Konto eine Methode `einzahlen`, die den auf das Konto einzuzahlenden Betrag als Parameter vom Datentyp `double` erhält. Die Einzahlung soll aber nur dann tatsächlich ausgeführt werden, wenn der übergebene Betrag positiv ist. Um dem Aufrufenden zu signalisieren, ob die Transaktion durchgeführt wurde, soll ein `boolean`-Wert zurückgegeben werden. Falls die Transaktion stattgefunden hat soll `true`, falls nicht `false` zurückgegeben werden.

d) Um auch wieder Geld auszahlen zu können benötigen Sie eine zweite Methode `auszahlen`. Auch diese erhält den auszuzahlenden Betrag als Parameter (Gleitkommazahl) und soll den Kontostand nur dann verringern, wenn der übergebene Betrag positiv ist und auf dem Konto noch ausreichend Geld vorhanden ist (der Kontostand darf nach der Auszahlung nicht negativ werden). Auch hier soll eine erfolgreiche Transaktion durch Rückgabe des `boolean`-Wertes `true` und eine nicht erfolgreiche durch `false` signalisiert werden.

e) Erstellen Sie eine neue Klasse `KontoVerwaltung`, die die *Main*-Methode enthält. Dort sollen für fünf Kunden der Bank je ein neues `Konto`-Objekt angelegt werden. Vergeben Sie die Kontonummern 111, 222, 333, 444, 555. Lassen Sie anschließend für alle fünf Konten die Kontonummer sowie den zugehörigen Kontostand auf der Konsole ausgeben.

f) Um ihr Programm zu testen, sollen die untenstehenden Buchungen auf den Konten mit den jeweiligen Kontonummern ausgeführt werden. Nutzen Sie hierzu die von Ihnen gewählten Bezeichner für die Konto-Objekte. Wenn eine Buchung fehlschlägt, geben Sie eine Fehlermeldung auf der Konsole aus: „Die Buchung konnte nicht ausgeführt werden“.

| Kontonummer | Einzahlung | Auszahlung |
|---|---|---|
| 111 | 500,00 € | |
| 222 | -5,00 € | |
| 333 | 46,50 € | |
| 444 | | 5,60 € |
| 111 | | 42,00 € |
| 555 | 4,33 € | |

**Beispiel:**
```
erstesKonto.einzahlen(500);  // Einzahlung auf das Konto 111
```

g) Lassen Sie nun ein zweites Mal für alle Konten die Kontonummer mit den entsprechenden Kontoständen ausgeben.

## Lösung Aufgabe 3

```
1   public class Konto {
2         // Aufgabenteil a)
3         int kontonummer;
4         double kontostand;
5
6         // Aufgabenteil b)
7         public Konto(int nr) {
8               kontonummer = nr;
9               kontostand = 0.0;
10        }
11
12        // Aufgabenteil c)
13        public boolean einzahlen(double betrag) {
14              if (betrag > 0) {
15                    kontostand = kontostand + betrag;
16                    return true;
17              } else {
18                    return false;
19              }
20        }
21
22        // Aufgabenteil d)
23        public boolean auszahlen(double betrag) {
24              if (betrag > 0 && kontostand >= betrag) {
25                    kontostand = kontostand - betrag;
26                    return true;
27              } else {
28                    return false;
29              }
30        }
31  }
```

```java
public class KontoVerwaltung {

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

		System.out.println("Kontostand für Kontonr. " + erstesKonto.kontonummer +
		": " + erstesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + zweitesKonto.kontonummer
		+ ": " + zweitesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + drittesKonto.kontonummer
		+ ": " + erstesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + viertesKonto.kontonummer
		+ ": " + viertesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + fuenftesKonto.kontonummer
		+ ": " + fuenftesKonto.kontostand);

		// Aufgabenteil f)
		System.out.println("Ein- und Auszahlungen werden getätigt...");
		if (!erstesKonto.einzahlen(500.00))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");
		if (!zweitesKonto.einzahlen(-5.00))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");
		if (!drittesKonto.einzahlen(46.50))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");
		if (!viertesKonto.auszahlen(5.60))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");
		if (!erstesKonto.auszahlen(42.00))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");
		if (!fuenftesKonto.einzahlen(4.33))
			System.out.println("Die Buchung konnte nicht ausgeführt werden");

		// Aufgabenteil g)
		System.out.println("Kontostand für Kontonr. " + erstesKonto.kontonummer +
		": " + erstesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + zweitesKonto.kontonummer
		+ ": " + zweitesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + drittesKonto.kontonummer
		+ ": " + drittesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + viertesKonto.kontonummer
		+ ": " + viertesKonto.kontostand);
		System.out.println("Kontostand für Kontonr. " + fuenftesKonto.kontonummer
		+ ": " + fuenftesKonto.kontostand);
	}
}
```

Die Ausgabe lautet:

```
Kontostand für Kontonr. 111: 0.0
Kontostand für Kontonr. 222: 0.0
Kontostand für Kontonr. 333: 0.0
Kontostand für Kontonr. 444: 0.0
Kontostand für Kontonr. 555: 0.0
Ein- und Auszahlungen werden getätigt...
Die Buchung konnte nicht ausgeführt werden (Einzahlung auf Konto 222, negativer Betrag)
Die Buchung konnte nicht ausgeführt werden (Auszahlung von Konto 444, Konto nicht gedeckt)
Kontostand für Kontonr. 111: 458.0
Kontostand für Kontonr. 222: 0.0
Kontostand für Kontonr. 333: 46.5
Kontostand für Kontonr. 444: 0.0
Kontostand für Kontonr. 555: 4.33
```