

Lernerfolgskontrolle (Java)

PPR

Nur zur Übung. Es gibt keinerlei Garantien, dass die tatsächliche Klausur den gleichen Umfang und Schwierigkeitsgrad hat. In der Klausur wird es 40 Punkte für den Rechnerteil und 50 im Programmiereteil geben. Die Klausur wird auf 90 Minuten konzipiert sein, ihr habt jedoch 120 Minuten Zeit.

Probeklausur

Musterlösung  
Stand: 12. Februar 2020

Aufgabe	Punkte	Erreichte Punkte
1	8	
2	7	
3	5	
4	9	
5	9	
6	4	
7	5	
8	10	
9	4	
10	11	
Summe	72	

**Aufgabe 1 (8 Punkte) Zahlensystem.**

1. (3 Punkte) Berechnen Sie 15-47 unter Verwendung der 8 Bit Zweierkomplementdarstellung. Die Rechenschritte müssen erkennbar sein.

**Lösung:**

$$15_{10} = 8_{10} + 4_{10} + 2_{10} + 1_{10} = 00001111_2$$

$$47_{10} = 32_{10} + 8_{10} + 4_{10} + 2_{10} + 1_{10} = 00101111_2$$

-47<sub>10</sub> bilden: 47<sub>10</sub> = 00101111<sub>2</sub> invertieren, → 11010000<sub>2</sub>, um eins inkrementieren, → -47<sub>10</sub> = 11010001<sub>2</sub>.

$$\begin{array}{r} 0|00001111 \\ + 1|11010001 \\ \hline \hline 1|11100000 \end{array}$$

1|1 → kein Über-/Unterlauf.

$$11100000_2 = -128_{10} + 64_{10} + 32_{10} = -32_{10}$$

2. (2 Punkte) Die genetischen Informationen sind bei allen Lebewesen in den DNA-Molekülen gespeichert. Ein DNA-Molekül ist eine lineare Sequenz der vier Nukleobasen A,C,G und T. Das menschliche Genom enthält rund  $2 \times 10^9$  Nukleobasen. Wie groß ist das gesamte Datenvolumen des menschlichen Genoms in Bytes?

**Lösung:**

$$(\log_2 4 * 2 * 10^9) / 8 = (2 * 2 * 10^9) / 8 = 500 * 10^6$$

3. (3 Punkte) Stellen Sie 30,25 in der 2 Byte binären Gleitkommadarstellung dar.  
(1 Vorzeichenbit, 4 Bit Exponent: 7-Exzess-Darstellung, 11 Bit Mantisse).

**Lösung:**

Fixkomma:  $30,25_{10} = 16_{10} + 8_{10} + 4_{10} + 2_{10} + 0,25_{10} = 11110,1_2$

Normieren:  $11110,1_2 = 1,11101_2 * 2^{4_{10}} \rightarrow e = 4_{10}$ , wegen 7-Exzess folgt  $\rightarrow E = 11_{10} = 8_{10} + 2_{10} + 1_{10} = 1011_2$ .

VZ-Bit: positiv  $\rightarrow 0$ , Mantisse: normierter Nachkommaanteil, aufgefüllt mit Nullen.  $\rightarrow 11101000000_2$ .

$\rightarrow 0\ 1011\ 11101000000$

**Aufgabe 2 (7 Punkte) Logische Schaltungen.**

1. (4 Punkte) Erstellen Sie eine logische Funktion, die anhand eines 3-stelligen Nachkommaanteils einer binären Festkommazahl ( $\dots, x_0x_1x_2$ ) über zwei Ausgänge ausgibt, ob auf- (ab .5) oder abgerundet werden sollte. Falls der Nachkommaanteil einer 0 entspricht, soll weder auf- noch abgerundet werden. Geben Sie die Wertetabelle sowie die Gatterschaltung an!

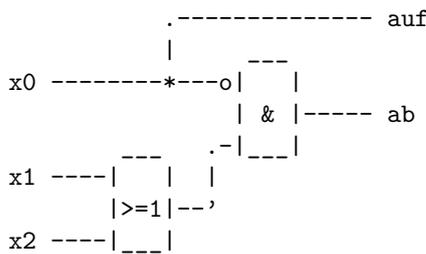
**Lösung:**

$x_0$	$x_1$	$x_2$	auf	ab
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Offensichtlich so:

$$ab = \overline{x_0}(\overline{x_1}x_2 + x_1\overline{x_2} + x_1x_2) = \overline{x_0}(x_1 + x_2)$$

$$auf = x_0(x_1 + x_2 + \overline{x_1}\overline{x_2}) = x_0$$



2. (3 Punkte)

Sie wollen einen Tabellenspeicher mit 32 Zeilen für einen 8-Bit Rechner bauen!  
Sie können die Ergebnisse qualitativ angeben z.B.  $w = 2^n - 1$

- (a) Wieviele BinaryCells benötigen Sie dazu insgesamt?  
**32\*8 = 256**
- (b) Mit welchem Bauteil wird die Adressierung der einzelnen Zeilen des Tabellenspeichers realisiert?  
**Demultiplexer**
- (c) Wie breit muss der Adressbus sein, um jede Zeile des Tabellenspeichers adressieren zu können?  
**5 Bit weil  $\log_2(32) = 5$  oder  $2^5 = 32$**

**Aufgabe 3 (5 Punkte) Rechnerarchitektur.**

1. (3 Punkte)

Was versteht man unter einem *Betriebssystem*? Nennen Sie *drei Aufgaben* eines Betriebssystems.

**Lösung:**

- (a) Der Start und die Verwaltung von Prozessen (laufenden Programmen)
- (b) Speichermanagement
- (c) Erzeugung und Verwaltung eines logischen Dateisystems

2. (2 Punkte)

Erläutern Sie kurz die Begriffe *RAM* und *ROM*. Nennen Sie 2 Typen für RAM.

**Lösung:**

RAM ist die Abkürzung für random access memory. Der Inhalt kann jederzeit verändert werden. Es gibt statischen RAM, bestehend aus Flip-Flops, typischerweise auf Grund seiner hohen Geschwindigkeit für Cache verwendet. Dynamischer RAM, aufgebaut aus Kondensatoren, wird auf Grund seiner preiswerteren Herstellung typischerweise für den Hauptspeicher eingesetzt.

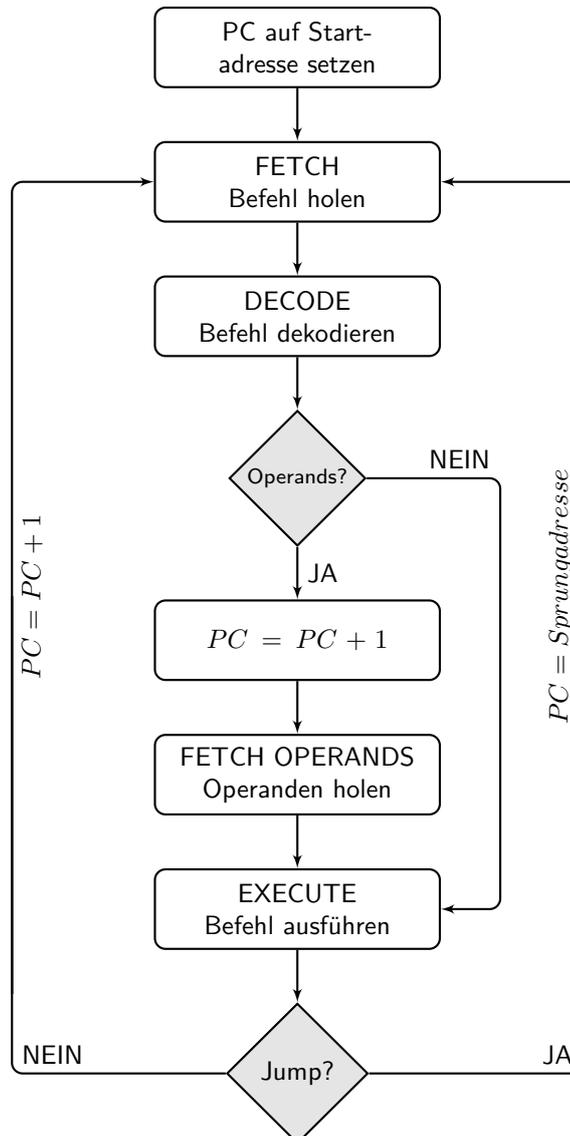
ROM bezeichnet read-only memory, dessen Inhalt fest "eingeschnitten" ist und deswegen nicht verändert werden kann.

**Aufgabe 4 (9 Punkte) Programmablauf im Rechner.**

1. (2 Punkte) Das folgende Blockschaltbild visualisiert den Befehlszyklus einer CPU. Tragen Sie die folgenden Begriffe an die passenden Stellen in das Blockschaltbild ein. Es werden nur Zuordnungen gewertet, die deutlich erkennbar sind.

- (a) PC+1
- (b) Sprungadresse
- (c) PC+1
- (d) EXECUTE - Befehl ausführen
- (e) FETCH - Befehl holen
- (f) FETCH OPERANDS - Operanden holen
- (g) PC auf Startadresse setzen
- (h) DECODE - Befehl dekodieren

**Lösung:**



2. (7 Punkte)

Die folgende Tabelle stellt einen Speicherausschnitt vor Beginn der darunter angegebenen Befehlsausführung dar. Der Index  $_2$  ist benutzt, um anzuzeigen, dass diese Zahl im Binärsystem angegeben ist.

RAM		Befehlsbeschreibungen:	
0135	01100110 <sub>2</sub>	CLA	Lädt AC mit 0.
0136	11110000 <sub>2</sub>	LDA M	Lädt AC mit dem Inhalt der Speicherzelle an Adresse M.
0137	00001100 <sub>2</sub>	STA M	Speichert den Inhalt von AC in der Speicherzelle an Adresse M.
...		ADD M	Addiert den Inhalt der Speicherzelle an Adresse M zum Inhalt von AC (und speichert wieder in AC).
1205	BR	SUB M	Subtrahiert den Inhalt der Speicherzelle an Adresse M vom Inhalt in AC (und speichert wieder in AC).
1206	1209	AND M	Bitweises AND des Inhalts der Speicherzelle an Adresse M mit dem Inhalt in AC.
1207	XOR	XOR M	Bitweises XOR des Inhalts der Speicherzelle an Adresse M mit dem Inhalt in AC.
1208	137	BR L	Lädt den Befehlszähler PC mit der Adresse L.
1209	XOR		
1210	136		
1211	STA		
1212	135		

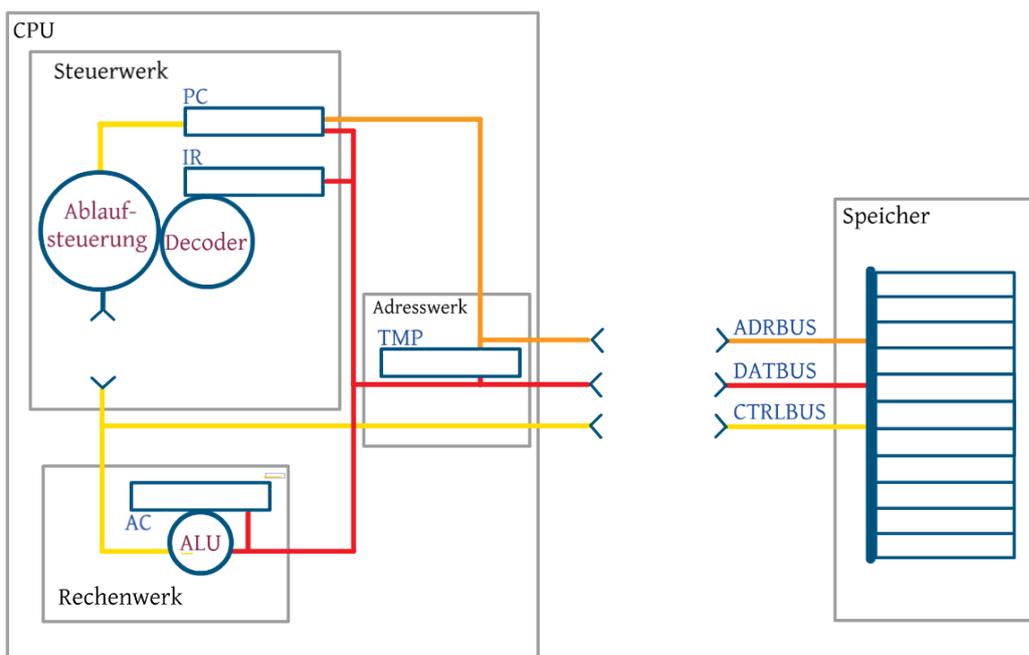
- Vervollständigen Sie nun die folgende Programmablaufstabelle! (Jede Zeile gibt die Belegung nach Ausführung der durch CTRL angegebenen Anweisungen an.)

**Lösung:**

PC	IR	AC	TMP	CTRL	CBUS	ABUS	DBUS
1205	BR	11001100 <sub>2</sub>	4321	PC++, PC→ABUS, RAM→DBUS, DBUS→IR	read	1205	BR
1206	BR	11001100 <sub>2</sub>	1209	PC++, PC→ABUS, RAM→DBUS, DBUS→TMP	read	1206	1209
1209	BR	11001100 <sub>2</sub>	1209	TMP→PC	-	1206	1209
1209	XOR	11001100 <sub>2</sub>	1209	PC→ABUS, RAM→DBUS, DBUS→IR	read	1209	XOR
1210	XOR	11001100 <sub>2</sub>	136	PC++, PC→ABUS, RAM→DBUS, DBUS→TMP	read	1210	136
1210	XOR	00111100 <sub>2</sub>	136	TMP→ABUS, RAM→DBUS, DBUS→ALU	read	136	11110000 <sub>2</sub>
1211	STA	00111100 <sub>2</sub>	136	PC++, PC→ABUS, RAM→DBUS, DBUS→IR	read	1211	STA
1212	STA	00111100 <sub>2</sub>	135	PC++, PC→ABUS, RAM→DBUS, DBUS→TMP	read	1212	135
1212	STA	00111100 <sub>2</sub>	135	TMP→ABUS, AC→DBUS, DBUS→RAM	write	135	00001100 <sub>2</sub>

1 Punkt für die vier BR-Felder; 1 Punkt für die vier 136-Felder; 1 Punkt für  $TMP \rightarrow PC$ ; 0,5 für alle anderen.

Das Blockschaltbild zur Erinnerung:



**Aufgabe 5 (9 Punkte) Betriebssystem.**

1. (3 Punkte)

Sie haben gerade die ersten PPR-Hausaufgaben in Ihr home-Verzeichnis heruntergeladen, aber sich leider nicht gemerkt wie die Datei heißt. Ihr home-Verzeichnis ist ziemlich überfüllt und unübersichtlich. Sie sind sich sicher, dass es eine pdf-Datei war. Geben Sie alle notwendigen Befehlsaufrufe für Folgendes an:

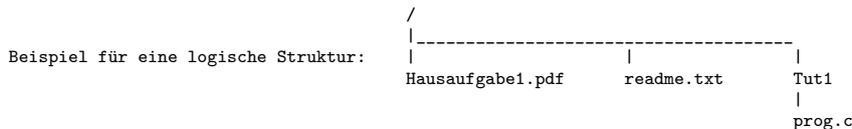
- Wechseln Sie in der Konsole erst in Ihr home-Verzeichnis
- Um die Datei zu finden, lassen Sie sich alle pdf-Dateien im Home-Verzeichnis anzeigen, inkl. Änderungsdatum.
- Sie haben nun **Hausaufgabe1.pdf** gefunden: erzeugen Sie jetzt ein Verzeichnis namens **PPR** und **verschieben** Sie die Datei dorthin.

**Lösung:**

```
cd ~
ls -l *.pdf
mkdir PPR
mv Hausaufgabe1.pdf PPR/
```

2. (6 Punkte)

Ermitteln Sie aus folgendem Betriebssystem-internem Zustand des Dateisystems die logische Struktur (Baum), die der Benutzer sieht.



Betriebssystem-interne Sicht Dateisystem:

*inode-Liste*

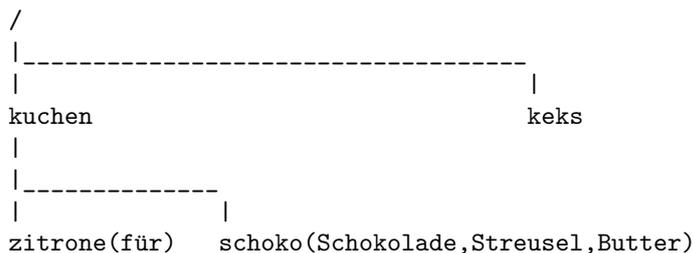
inode-Id	Dateityp	Speicheradresse(n)
2	d	1-2,22,7
3	-	19
303	d	4-5
304	-	16-18
500	d	23-25,3

*Datenblock*

Speicheradresse	Inhalt
1	“.“ id 2
2	“.“ id 2
3	“zitrone“ id 3
4	“.“ id 2
5	“.“ id 303
7	“keks“ id 303
16	Schokolade
17	Streusel
18	Butter
19	für
22	“kuchen“ id 500
23	“.“ id 500
24	“.“ id 2
25	“schoko“ id 304

Logische Sicht Dateisystem:

**Lösung:**



**Aufgabe 6 (4 Punkte) Prozesse.**

Beantworten Sie durch Ankreuzen von „korrekt“ oder „falsch“, ob die folgenden Aussagen richtig sind. Für jedes richtig gesetzte Kreuz gibt es einen halben Punkt.

	<i>korrekt</i>	<i>falsch</i>
Sie haben gedit im Hintergrund geöffnet. Dieser Prozess muss im Zustand "running" sein.	( )	(x)
Unter Unix kann man als Programmierer sicherstellen, dass das eigene Programm genau so oft Rechenzeit bekommt, wie man das möchte, z.B. alle 100ms.	( )	(x)
Der Nutzer kann die Priorität von Programmen beeinflussen. Systemprogramme haben eine höhere Priorität.	(x)	( )
Ein negativer nice-Wert bedeutet, dass dieser Prozess die höchste Priorität hat. Ein Prozess, der viel Rechenzeit benutzt, hat eine höhere Priorität.	( )	(x)
Ein Prozess, der länger wartet, hat eine höhere Priorität. Von 2 ansonsten gleichen Prozessen hat der mit dem niedrigerem nice-Wert die höhere Priorität.	(x)	( )

**Aufgabe 7 (5 Punkte) Java: Einführung.**

1. (3 Punkte)

Beantworten Sie durch Ankreuzen von „korrekt“ oder „falsch“, ob die folgenden Aussagen richtig sind. Für jedes richtig gesetzte Kreuz gibt es einen halben Punkt.

	<i>korrekt</i>	<i>falsch</i>
<code>public double int;</code> deklariert eine Kommazahl-Variable.	( )	(x)
Bei Bezeichnern in Java wird Groß- und Kleinschreibung nicht unterschieden.	( )	(x)
Interfaces sind Instanzen von Klassen.	( )	(x)
Die Funktion <code>public void getValue( int a )</code> gibt einen String zurück.	( )	(x)
Der Schleifenrumpf von <code>for ( int i = 10; i &gt;= 0; i-- ){...}</code> wird 10 Mal ausgeführt.	( )	(x)
Objekt und Objektvariable ist nicht das gleiche.	(x)	( )

2. (2 Punkte) Schreiben Sie für folgende Funktionalitäten jeweils einen passenden *Methodenkopf* auf.

- (a) die Methode gibt die Geschwindigkeit zurück, berechnet anhand der übergebenen Strecke (m) und Zeit (s).
- (b) die Methode gibt alle Primzahlen zwischen zwei übergebenen beliebigen Zahlen auf dem Bildschirm aus.
- (c) die Methode gibt zurück, ob eine übergebene Zahl eine Primzahl ist.

- (a) `public static float speed( float distance_m, float time_s );`
- (b) `public static void printPrimes( int a, int b );`
- (c) `public static boolean isPrime( int number );`

---

**Aufgabe 8 (10 Punkte) Java.**

1. (5 Punkte) Führen Sie eine Handsimulation durch.

- Tragen Sie die Variablenbelegungen, **nach Ausführung der Zeile** in die Tabelle ein. Achten Sie dabei auf die korrekte Darstellung des jeweiligen Typs.
- Kennzeichnen Sie aktuell nicht existierende Variablen mit `-`.
- Kennzeichnen Sie existierende Variablen, die noch nicht mit einem Wert belegt sind mit `undef`.
- Schreiben Sie bei Methodenaufrufen zusätzlich die aufrufende Zeile in Klammern hinter die momentane Zeilennummer. Bsp: Wir befinden uns in Zeile 14 in einer Funktion, die in Zeile 34 aufgerufen wurde. **Zeile: 14 (34)**
- Beachten Sie bei Methodenaufrufen, dass die erste Zeile der aufgerufenen Methode der zugehörige Methodenkopf ist.
- Nach Aufruf eines `return`-Statements ist die nächste ausgeführte Zeile der Methodenabschluss `}`.
- Die letzte Zeile einer Methode ist immer die schließende Klammer `}`.
- Die Zahl der Zeilen in der Tabelle ist abgezählt, d.h., so viele Zeilen werden im Code durchlaufen und von euch bitte beschrieben.

```

1 public class Handsimulation{
2     public static int mod(int x, int y){
3         int z = x / y;
4         if( z == 0 ){
5             return 0;
6         }
7         return x-(z*y);
8     }
9     public static void sub(int x){
10        x=x-1;
11    }
12    public static void main(String[] args){
13        int x = 3;
14        sub(x);
15        int erg = mod(x,2);
16        boolean b = (erg == x);
17    }
18 }

```

Zeile	x(main)	erg	b	x(sub)	x(mod)	y	z
12	-	-	-	-	-	-	-
13	3						
9(14)				3			
10(14)				2			
11(14)				-			
14							
2(15)					3	2	
3(15)							1
4(15)							
6(15)							
7(15)							
8(15)					-	-	-
15		1					
16			False				
17	-	-	-				

2. (2 Punkte)

```
public static void f(boolean a, boolean b) {
    boolean c;
    if ( !(a || b) && !(b && !a) ){
        c = true;
    }else{
        c = false;
    }
    System.out.println(c);
}
```

Geben Sie in der folgenden Tabelle an, was die Methode *f* bei der jeweiligen Belegung der Parameter ausgibt.

a	b	Ausgabe von f(a,b)
false	false	<i>true</i>
false	true	<i>false</i>
true	false	<i>false</i>
true	true	<i>false</i>

3. (3 Punkte) Schreiben Sie eine Methode `oddNumber`, die ein Array von Ganzzahlen übergeben bekommt und jede Zahl des Arrays ausgibt, falls sie ungerade ist. Beachten Sie, dass der Methode `oddNumber` beliebig lange Arrays übergeben werden könnten.

**Hinweis:** Gegeben sei eine Klasse `FunMath`, die eine statische Methode `isEven(int x)` beinhaltet. Die statische Methode `isEven(int x)` gibt *true* zurück, wenn es sich um eine gerade Zahl handelt, andernfalls gibt sie *false* zurück.

```
public void oddNumber(int[] zahlen){
    for(int i = 0; i<zahlen.length; i++){
        if(!FunMath.isEven(zahlen[i])){
            System.out.println(zahlen[i]);
        }
    }
}
```

**Aufgabe 9 (4 Punkte) Java.**

Gegeben seien folgende Java-Klassen:

```

01  class X {
02      static int s = 2;
03      void f1(){
04          System.out.println("X::f1");
05      }
06      void f2(){
07          System.out.println("X::f2");
08      }
09  }
10
11  class Y extends X {
12      static int s = 1;
13      void f1(){
14          System.out.println("Y::f1");
15      }
16      void f3(){
17          System.out.println("Y::f3");
18      }
19  }
20
21  public class Test {
22      public static void main (String[ ] args) {
23          X x = new Y();
24          Y y = new Y();
25          /***/
26      }
27  }

```

Geben Sie an, was die Ausgabe der folgenden Anweisungen ist, wenn sie anstelle des Kommentars in Zeile 25 eingefügt werden. Neben der Ausgabe sind auch *Compilierfehler* bzw. *Laufzeitfehler* mögliche Antworten.

**Lösung:**

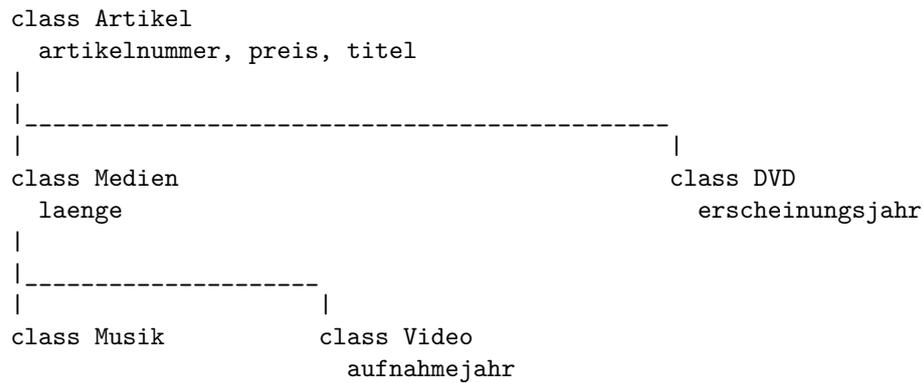
1. ((Y) x).f1();                      \\ Y::f1
2. ((X) y).f1();                      \\ Y::f1
3. System.out.println(x.s);            \\ 2
4. y.f3();                            \\ Y::f3
5. y.f2();                            \\ X::f2
6. y.f1();                            \\ Y::f1
7. Y z = new X(); z.f1();            \\ Compilerfehler
8. X z = new X(); ((Y) z).f1();      \\ Laufzeitfehler

## Aufgabe 10 (11 Punkte) Java.

1. (2 Punkte) Klaus verkauft Artikel bei Ebay:

- *Musik*: Beschrieben durch Artikelnummer, Preis, Titel und Länge
- *Videokassette*: Beschrieben durch Artikelnummer, Preis, Titel, Länge und Aufnahmejahr
- *DVD*: Beschrieben durch Artikelnummer, Preis, Titel und Erscheinungsjahr

Stellen Sie die Vererbungshierarchie grafisch dar. Verwenden Sie (wenn nötig) auch zusätzliche Klassen um Redundanzen zu vermeiden.

**Lösung:**

2. (6 Punkte) Betrachten Sie folgende Klassen und das dazugehörige Interface. Das Uboot und das Flugzeug implementieren beide das Interface Fahrzeug. Die Attribute von Uboot und Flugzeug sind nur innerhalb der Klasse sichtbar. Befüllen Sie die Lücken und implementieren Sie die unten stehende Testklasse anhand der gegebenen Kommentare.

```
public interface Fahrzeug{
    public void beschleunigen();
}
```

**Lösung:**

```
public class Uboot implements Fahrzeug {

    private double knoten; // Geschwindigkeit

    public Uboot(double k){
        this.knoten = k;
    }
    // Methode beschleunigen
    public void beschleunigen(){
        knoten++;
    }
}
```

```
public class Flugzeug implements Fahrzeug {

    private double kmh; // Geschwindigkeit des Flugzeugs

    // Methode beschleunigen
    public void beschleunigen(){
        kmh= ((int)Math.random()*100)*kmh;
    }
}
```

```
public class TestFahrzeug {
    public static void main (String[] args){
        // Erzeugen Sie eine ArrayListe vom Typ List
        List<Fahrzeug> fzgListe = new ArrayList<Fahrzeug>();

        // Fügen Sie ein Uboot zur Liste hinzu
        fzgListe.add( new Uboot( 30 ) );

        // Fügen Sie ein Flugzeug zur Liste hinzu
        fzgListe.add( new Flugzeug() );

        // Geben Sie die Laenge der Liste auf dem Bildschirm aus.
        System.out.println( "Die Fahrzeugliste hat " + fzgListe.size() + " Elemente." );
    }
}
```

3. (3 Punkte)

Beantworten Sie durch Ankreuzen von „korrekt“ oder „falsch“, ob die folgenden Zeilen korrekten Java-Code darstellen. Jede richtige Antwort gibt 0,5 Punkte, für jede falsche Antwort werden 0,5 Punkte abgezogen. Insgesamt können aber nicht weniger als 0 Punkte erreicht werden.

	<i>korrekt</i>	<i>falsch</i>
<code>boolean b = Fahrzeug instanceof Uboot;</code>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<code>Fahrzeug fz = new Fahrzeug();</code>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<code>Flugzeug ub = new Uboot();</code>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<code>Flugzeug uf = new Fahrzeug();</code>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<code>Fahrzeug ub = new Uboot();</code>		
<code>ub.beschleunigen();</code>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<code>Fahrzeug uf = new Flugzeug();</code>		
<code>uf.beschleunigen();</code>	<input checked="" type="checkbox"/>	<input type="checkbox"/>



