

Name, Vorname:

Hilfsmittel:  
keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
- In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

## AUFGABEN

1) 2P

Welche Zuweisungen sind korrekt bzw. nicht korrekt ?

- a) "Objekt einer Unterklasse" = "Objekt der Oberklasse"
- b) "Objekt einer Oberklasse" = "Objekt der Unterklasse"

2) 12P

- a) Was sind abstrakte Klassen und welchen Sinn haben sie (mit Beispiel) ?
- b) Was sind abstrakte Methoden und welchen Sinn haben sie (mit Beispiel)?
- c) Was sind Klassenvariable und welchen Sinn haben sie (mit Beispiel)?

Bemerkung: Bei den Beispielen keine Programme oder Programmcode bringen.

3) 36P

(Alle Teilaufgaben müssen in **einem** Programm realisiert werden)

**Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.**

a) Erstellen Sie die Klasse Henne, (mit genau 3 Attributen ,wobei 2 Attribute davon "name" und "nummer" heißen müssen, genau 2 set-Methoden, genau 3 get-Methoden und genau 1 Konstruktor), die eine Henne mit einem Namen und einer Nummer repräsentieren soll. Die später zu erzeugenden Hennen sollen alle automatisch (beim Erzeugen der jeweiligen Henne) durchnummeriert werden. D.h. die erste erzeugte Henne bekommt die Nummer 1, die zweite erzeugte Henne bekommt die Nummer 2, usw. ohne daß diese Nummer als Parameter übergeben wird.

- b) Erzeugen Sie ein Henne mit dem Namen "Berta"
- c) Verändern Sie deren Namen zu "Henne\_Berta".
- d) Erzeugen Sie ein Henne mit dem Namen "Ute"
- e) Verändern Sie deren Namen zu "Henne\_Ute".
- f) Geben Sie den Namen und die Nummer der 1. erzeugten Henne auf dem Bildschirm aus. (Bitte entsprechende Methoden benutzen).
- g) Geben Sie den Namen und die Nummer der 2. erzeugten Henne auf dem Bildschirm aus.
- h) Geben Sie die Hennenanzahl. d.h. die Anzahl der erzeugten Hennen auf dem Bildschirm aus. (Bitte entsprechende Methoden benutzen).

Lösungen:

1)

// 2P

"Objekt einer Oberklasse" = "Objekt der Unterklasse"

2)

// 12P

a)

Von manchen Klassen macht es keinen Sinn, ein Objekt davon zu erzeugen, d.h. man soll kein Objekt davon erzeugen können.

Beispiel: Es gibt in der realen Welt kein Nutztier, sondern nur spezielle Unterarten, wie z.B. Henne oder Kuh.

b)

Eine abstrakte Methode in der Oberklasse definiert lediglich den Methodenkopf und drückt damit aus, dass die Oberklasse keine Ahnung von der Implementierung hat und die jeweilige Unterklasse sich darum kümmern muss.

Beispiel:

Die abstrakte Methode "getTierwert()" in der Oberklasse Nutztier definiert den Methodenkopf. In den Unterklassen Henne oder Kuh muß die Methode dann ausprogrammiert werden.

c)

Eine Klassenvariablen bezeichnet eine Variable, die die ganze Klasse betrifft und nicht nur ein einzelnes Objekt.

Beispiel: Der Eierpreis auf dem Wochenmarkt ist für jede Henne gleich.



*Name, Vorname:*

Hilfsmittel:  
keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
- In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1)

6P

a)

Was versteht man unter dem Begriff "Vererbung" ?

Welchen Vorteil hat man beim Vererben ?

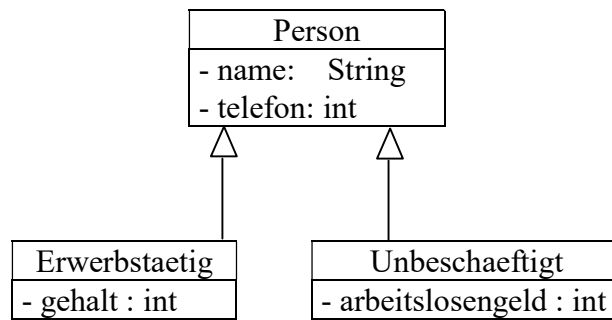
b)

Was versteht man unter einem verdeckten Member ?

2)

45P

Gegeben ist das folgende abgespeckte UML-Diagramm:



a)

Erzeugen Sie aus dem folgenden UML-Diagramm die entsprechenden Klassen mit den jeweiligen Konstruktoren, set- und get-Methoden.

Die Konstruktoren müssen jeweils Parameter enthalten.

b)

Erzeugen Sie in der Methode `main` (jeweils durch eine Anweisung) den Erwerbstätigen "Schaffer" mit der Telefonnummer 4711, dem Gehalt von 1500 Euro und den Unbeschäftigten "Schröder" mit der Telefonnummer 4712 und Arbeitslosengeld von 400 Euro.

c)

Herr Schaffer bekommt 100 Euro mehr Gehalt.

Realisieren Sie dies durch genau eine einzige Anweisung (keine neue Methode implementieren), die zum alten Gehalt die 100 Euro dazu addiert.

In der Anweisung muss also der alte Gehalt ermittelt werden.

d)

Herr Schröder meldet sein Telefon ab. Dies wird so realisiert, daß er die Telefonnummer - 1 bekommt.

Realisieren Sie dies durch genau eine einzige Anweisung.

e)

Geben Sie die Anweisung an, die den neuen Gehalt von H. Schaffer auf dem Bildschirm ausgibt.

Lösung:

1) 6P

a) 3P

Da es viel Schreibaufwand macht, die gleichen Methoden für die Manipulation eines Attributs (wie z.B. dem Alter und dem Namen) zu implementieren, ist es geschickter dies nur einmal in einer eigenen Klasse zu machen und diese Methoden dann in die Unterklassen zu vererben.

b) 3P

Wenn eine Methode sowohl in der Oberklasse als auch in der Unterklasse vorkommt, wird bei einem Zugriff auf diese Methode die Methode der Unterklasse zugegriffen.

2) 45P

```
public class Main_E2FI_8_4_08_nr1 {  
    public static void main(String[] args){  
        Erwerbstaetig e = new Erwerbstaetig("Schaffer", 4711, 1500);    // 3P  
        Unbeschaeftigt u = new Unbeschaeftigt("Schroeder", 4712, 400); // 3P  
        e.setGehalt(e.getGehalt()+100); // 3P  
        u.setTelefon(-1); // 2P  
        System.out.println("neuer Gehalt = "+e.getGehalt()); // 2P  
    }  
}
```

```
class Person{ // 14P  
    private String name;  
    private int telefon;  
  
    public Person (String pName, int pTelefon){  
        name = pName;  
        telefon = pTelefon;  
    }  
  
    public void setTelefon(int pTelefon){  
        telefon = pTelefon;  
    }  
  
    public void setName(String pName){  
        name = pName;  
    }  
  
    public int getTelefon(){  
        return(telefon);  
    }  
  
    public String getName(){  
        return(name);  
    }  
}
```

```
class Unbeschaeftigt extends Person{ // 9P
    private int arbeitlosengeld ;

    public Unbeschaeftigt(String pName, int pTelefon,
                           int pArbeitslosengeld){
        super(pName, pTelefon);
        arbeitlosengeld = pArbeitslosengeld;
    }

    public void setArbeitslosengeld(int pArbeitslosengeld){
        arbeitlosengeld = pArbeitslosengeld;
    }

    public int getArbeitslosengeld(){
        return(arbeitslosengeld);
    }
}

class Erwerbstaetig extends Person{ // 9P
    private int gehalt ;

    public Erwerbstaetig(String pName, int pTelefon, int pGehalt){
        super(pName, pTelefon);
        gehalt = pGehalt;
    }

    public void setGehalt(int pGehalt){
        gehalt = pGehalt;
    }

    public int getGehalt(){
        return(gehalt);
    }
}
```

*Name, Vorname:*

Hilfsmittel:  
keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
- In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

## AUFGABEN

1) 29P

Für die Verwaltung des Fuhrparks des Gega-Konzerns soll ein objektorientiertes Programm erstellt werden. Die objektorientierte Analyse ergab folgende Begriffe:

Auto, Benzinmotor, Dieselmotor, Fahrrad, Fahrzeug, Kfz, Motor, Motorrad

a)

Stellen Sie die Zusammenhänge zwischen diesen Begriffen in einem UML-Klassendiagramm dar. Auf die Angabe von Attributen und Methoden ist hierbei zu verzichten.

Evtl. vorhandene abstrakte Klassen sind in geeigneter Weise zu kennzeichnen.

Achten Sie auf die richtige Darstellung der Beziehungen und geben Sie die Kardinalitäten an.

b) 21P

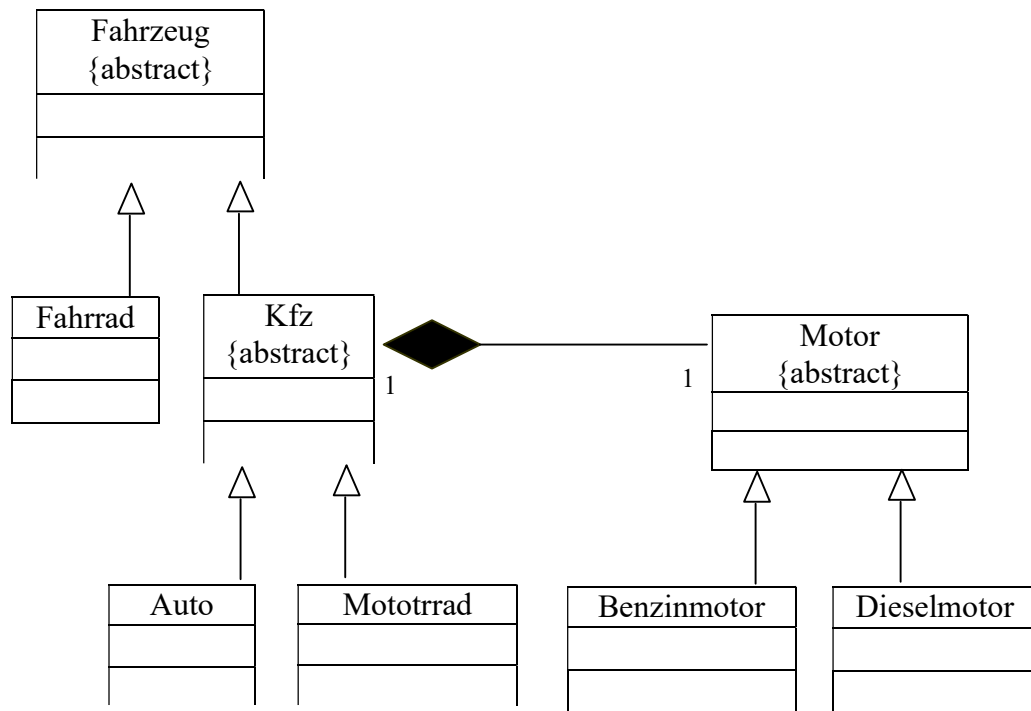
Erstellen Sie den Quellcode für die Klasse Motor. Geben Sie dabei zwei sinnvolle Attribute und die dafür mindestens erforderlichen Methoden an. Geben Sie zusätzlich noch genau einen Konstruktor an (kein Standardkonstruktor).

Erstellen Sie ein Objekt der Klasse Motor.

Nachdem ein Fahrer das Auto 24 Stunden im Vollgas gefahren hat, hatte der Motor den Kolbenfresser und ist kaputt gegangen. Bilden Sie dies durch eine entsprechende Java-Anweisung ab.



a)



b)

```
package e3fs_7_3_16;
```

```
public class Startklasse {
    public static void main(String[] args) {
        Motor myMotor = new Motor(1000, 100); // 3P
        myMotor = null;                       // 1P
    }
}
```

```
class Motor {
    private double hubraum; // 1P
    private double leistung; // 1P

    Motor(double hubraum, double leistung) { // 3P
        this.hubraum=hubraum;
        this.leistung=leistung;
    }

    public double getHubraum() { // 3P
        return hubraum;
    }

    public void setHubraum(double hubraum) { // 3P
        this.hubraum = hubraum;
    }

    public double getLeistung() { // 3P
        return leistung;
    }

    public void setLeistung(double leistung) { // 3P
        this.leistung = leistung;
    }
}
```

