

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) 14P

- a) Was ist eine Klasse ?
- b) Was ist ein Objekt?
- c) Was enthält eine Klasse?
- d) Was ist der Sinn einer Methode?
- e) Was ist ein Attribut?
- f) Was bedeutet private?
- g) Was bedeutet public?

2) 4P

gegeben sei folgender Programmausschnitt (Hund ist eine Klasse):

```
...  
Hund myh1;  
myh1 = new Hund();  
...
```

Was veranlassen diese zwei Zeilen Programmcode im Arbeitsspeicher?

Bezeichnung Adresse Inhalt

myh1	0800	

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

32P

a) Erstellen Sie die Klasse Punkt, (mit den entsprechenden Attributen, Methoden und zwei Konstruktoren), die einen Punkt (Pixel) in einer grafischen zweidimensionalen Oberfläche repräsentieren soll.

b) Erzeugen Sie einen Punkt mit der x-Koordinate 5 und der y-Koordinate 10.

c) Verändern Sie die Koordinaten um den Wert $+u$ in x- und den Wert $-u$ in y-Richtung, wobei u eine Variable ist, deren Wert dem Programmierer unbekannt ist.

Insbesonderte darf also nicht davon ausgegangen werden, daß die x-Koordinate 5 und die y-Koordinate 10 ist.

d) Ermitteln Sie mit Hilfe der entsprechenden Methoden die neuen Koordianten des Punktes und geben diese auf dem Bildschirm aus.

e) Erzeugen Sie einen anderen, neuen Punkt mit der x-Koordinate 6 und der y-Koordinate 11.

Lösung:

1) 14 Punkte

- a) Ein Bauplan, nach dem ein Objekt erstellt wird
- b) Ein nach einem Bauplan im Arbeitsspeicher angelegter, reservierter Speicher.
- c) Attribute und Methoden.
- d) Attribute zu lesen und zu beschreiben.
- e) Daten in einer Klasse.
- f) Auf private Member darf nur innerhalb einer Klasse zugegriffen werden.
- g) Auf public Member darf innerhalb und außerhalb einer Klasse zugegriffen werden.

2) 4 Punkte

Bezeichnung	Adresse	Inhalt
myh1	0800	0900
	...	
	0900	Attribute
		des
		Objekts

3)

```
public class MainKlassen6 {
    public static void main(String[] args) throws Exception{
        Punkt p1, p2;
        // 3 P
        p1= new Punkt(5,10);
        // 3 P
        p1.setX(p1.getX()+u);
        p1.setY(p1.getY()-u);
        // 3 P
        System.out.println("x= "+p1.getX());
        System.out.println("y= "+p1.getY());
        // 3 P
        p2= new Punkt(6,11);
    }
}
```

```
class Punkt{
    private double x; // 1P
    private double y; // 1P

    public Punkt(){ // 3P
        x=0;
        y=0;
    }

    public Punkt(double px, double py){ // 3P
        setPunkt(px,py);
    }

    public void setX (double px){ // 3P
        x=px;
    }

    public void setY (double py){ // 3P
        y=py;
    }

    public double getX(){// 3P
        return(x);
    }

    public double getY(){// 3P
        return(y);
    }
}
```

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

I)

1)

22P

Entwickeln Sie ein Programm, das den Umfang und den Flächeninhalt von Kreisen berechnet. Das Programmdesign und die Lösung soll **objektorientiert** sein.

$$U = 2\pi r \quad \text{und} \quad A = \pi r^2$$

Bemerkung:

In der Klasse Kreis dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen.

a) 22P

Erzeugen Sie die Klasse Kreis mit genau dem Attribut (und nur dem Attribut) "radius", den den zu "radius" gehörigen get-und set-Methoden und 2 Konstruktoren. Erzeugen Sie die Methode berechneFlaeche und berechneUmfang.

2)

28P

Realisieren Sie (mit Hilfe der entsprechenden Methode der Klasse Rechteck) Folgendes in der Methode main(..) der Startklasse:

a) 4P

Erzeugen Sie in main zwei Kreise:
k1 mit Radius 2 und k2 mit Radius 10.

b) 6P

Geben Sie die Daten (Radius, Umfang, Fläche) von k2 (durch Verwendung der entsprechende(n) Methoden) auf dem Bildschirm aus.

c) 6P

Aus Testgründen wird der über Tastatur eingegebene Radius des Objekts k1 verdoppelt und in dem Objekt k2 gespeichert. Dies soll in **einem** Ausdruck geschehen.

d) 6P

Geben Sie die Daten (Radius, Umfang, Fläche) von k_2 (durch Verwendung der entsprechende(n) Methoden) nach der Verdopplung aus.

e) 6P

Berechnen Sie (durch Verwendung der entsprechende(n) Methoden), um das Wievielfache sich dann der Flächeninhalt erhöht hat und geben Sie dieses Ergebnis auf dem Bildschirm aus.

Lösung:

```
1)
class Kreis{
    private double radius;                // 2P
    public static final double pi = 3.14;

    public Kreis(double pRadius){        // 2P
        radius = pRadius;
    }

    public Kreis(){                       // 2P
    }

    public void setRadius(double pRadius){ // 4P
        radius = pRadius;
    }

    public double getRadius(){            // 4P
        return(radius);
    }

    public double berechneUmfang(){        // 4P
        return(2*pi*radius);
    }

    public double berechneFlaeche(){       // 4P
        return(pi*radius*radius);
    }
}

2)
public class MainTest4 {
    public static void main(String[] args){
        double verhaeltnis;
        // Teilaufgabe a)
        Kreis k1 = new Kreis(2);           // 2P
        Kreis k2 = new Kreis(10);          // 2P
        // Teilaufgabe b)
        System.out.println("k2.getRadius()= "+k2.getRadius()); // 2P
        System.out.println("k2.berechneUmfang()="+k2.berechneUmfang()); //2P
        System.out.println("k2.berechneFlaeche()="+k2.berechneFlaeche()); //2P
        // Teilaufgabe c)
        k2.setRadius(k1.getRadius()*2);    // 6P
        // Teilaufgabe d)
        System.out.println("k2.getRadius()= "+k2.getRadius()); // 2P
        System.out.println("k2.berechneUmfang()="+k2.berechneUmfang()); // 2P
        System.out.println("k2.berechneFlaeche()="+k2.berechneFlaeche()); //2P
        // Teilaufgabe e)
        verhaeltnis = k2.berechneFlaeche()/k1.berechneFlaeche(); // 4P
        System.out.println("verhaeltnis= "+verhaeltnis); // 2P
    }
}
```

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) 12P

a)

Was versteht man unter dem Begriff "Vererbung" ?

Welchen Vorteil hat man beim Vererben ?

b)

Was versteht man unter einem verdeckten Member ?

c)

Was geschieht, wenn beim Aufruf einer Methode ein Wert benutzt wird, der nicht dem erwarteten Datentyp entspricht ?

d)

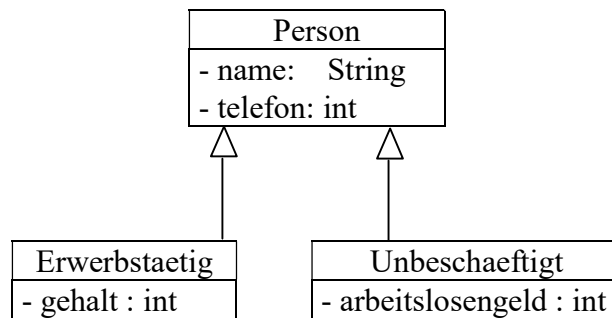
Wann unterscheiden sich die Spezifizierer public und protected ?

Kein Programm angeben, sondern eine verbale Beschreibung geben.

2)

39P

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, die zum alten Gehalt die 100 Euro dazu addiert.

In der Anweisung muss also der alte Gehalt ermittelt werden.

Lösung:

12P

1)

a) 3P

Alle Member einer Klasse (außer dem Konstruktor) werden an die erbende Klasse weitervererbt, d.h. können dort benutzt werden.

b) 2P

Damit können Softwareteile wiederverwendet werden und müssen nicht nochmals implementiert werden (Redundanzveringerung).

b) 3P

Eine Methode gleichen Namens existiert in der Ober- und Unterklasse.

Die Methode der Oberklasse wird durch die Methode der Unterklasse verdeckt, d.h. beim Aufruf in der Unterklasse wird die Methode der Unterklasse verwendet.

c) 3P

Bei der Vererbung: Der Datentyp der Unterklasse wird in den der Oberklasse konvertiert.
Sonst: Fehlermeldung des Compilers.

d) 1P

Wenn eine Methode weder in der gleichen Klasse, noch einer Unterklasse, noch im gleichen Package sind, dann kann auf diese Methode (im Gegensatz zu public) bei protected nicht zugegriffen werden.

2)

39P

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

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 Punkteabzug !!!!

AUFGABEN

1)

a) Erstellen Sie die Klasse "Bruch", die Brüche der Form a/b verwaltet, wobei a und b ganze Zahlen sind. Die Klasse soll die Methoden enthalten, mit denen man Brüche kürzt, addiert, subtrahiert, multipliziert und dividiert.

b) Berechnen Sie damit:

$$2/3 + 3/4$$

KLAUSUR 2 SAE E2FI Nachtermin2 Zeit: 45 Minuten

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 !!!!

1) Erstellen Sie die Klasse "Zufall", die durch den Aufruf von rand() eine zufällige Zahl aus einem bestimmten Bereich ausgibt.

Ein möglicher Algorithmus, um eine Folge von Zufallszahlen zu erzeugen, ist durch folgende Formel gegeben:

$$x_{n+1} = (a \cdot x_n + b + 1) \bmod n$$

Dabei sind a , b und n feste Zahlen. Für diese sind zunächst folgende Werte festgesetzt:

a= 1366

b = 113

m = 6

Der Anfangswert x_0 kann z.B. durch den Konstruktor festgelegt werden.

a) Erstellen Sie 10 Zufallszahlen und geben dann diese auf dem Bildschirm aus.

b) Versuchen Sie auch eine Methode bereitzustellen, die einen Münzwurf simuliert.

Bemerkung:

mod bedeutet den Rest bei der Division.

$$23 \bmod 5 = 3$$

weil 5 in 23 4 mal hineinpasst und dabei ein Rest von 3 übrig bleibt.

2) Erstellen Sie die Klasse Lottozahlen, die 6 zufällige ganze Zahlen zwischen 1 und 49 (je einschließlich) erzeugt (wobei diese - der Vereinfachung wegen - nicht notwendig verschieden sein müssen).

Diese Klasse ist eine spezielle Klasse der Klasse „Zufall“.

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) 13P

Es soll das Maximum (die größte Zahl) eines Integer-Feldes der Länge LEN berechnet werden. Erstellen Sie dazu das entsprechende Struktogramm.

2)

Gegeben ist das folgende abgespeckte, **fest vorgegebene** UML-Digramm.

Henne
- antibiotikaMenge: int

Kuh
- antibiotikaMenge: int

Auto
- maxSpeed : double

a) 12P

Implementieren Sie (unter Zuhilfenahme des obigen UML-Diagramms) die Klassen Henne und Auto (Klasse Kuh bitte nicht implementieren: sie wird analog Klasse Henne implementiert!) mit jeweils 2 Konstruktoren und den set- und get-Methoden.

Keine weiteren Attribute einfügen!

Zu Testzwecken sollen in den einzelnen Klassen die Methode void printAll() implementiert werden, die die Namen der Attribute mit den zugehörigen Werten auf dem Bildschirm ausgibt.

b) Das Programm soll multipersonal entwickelt werden: 17P
In einem Feld sollen verschiedene Objekte (Henne, Kuh, Auto) abgespeichert werden.
Danach sollen die Werte der Attribute mit der Methode `printAll()` ausgegeben werden.

b1) Ergänzen Sie das obige UML-Diagramm und die bisherige Implementierung, damit dies erreicht werden kann? (Unbedingt sinnvolle Namen geben!!).
Bitte keine Attribute der bereits bestehenden Klassen verändern oder neue einfügen!

b2) Was muß programmtechnisch gemacht werden, damit man die Entwickler der einzelnen Klassen zwingen kann, unbedingt die obige Methode `void printAll()` zu implementieren?

b3) Machen Sie folgendes in der Methode `main()`
- Erzeugen Sie die Henne "berta", die Kuh "milka" und das Auto "ente"
- Speichern Sie diese 3 Objekte in dem Feld.
- Geben Sie mit Hilfe einer Schleife und der Methode `printAll()` die Eigenschaften der jeweiligen Objekte auf dem Bildschirm aus.

c) 8P
In dem Feld "stall" sollen die im vorigen Programmteil erstellte Henne und Kuh abgespeichert werden und dann mit Hilfe einer Schleife und der Methode `getAntibiotikaMenge()` die Menge des gespritzten Antibiotikas des jeweiligen Tiers auf dem Bildschirm ausgegeben werden.

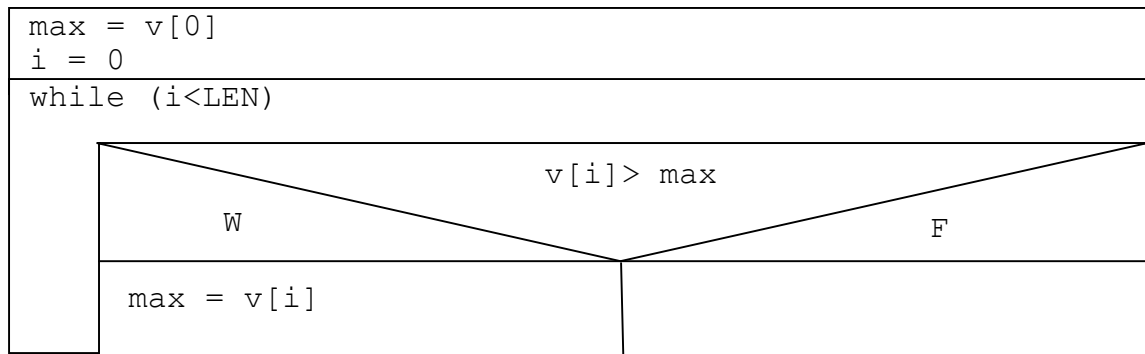
c1) Was wäre - vom Design her betrachtet - der Nachteil der Lösung, bei der man die Methode `getAntibiotikaMenge()` in einer Oberklasse von Henne, Kuh und Auto implementiert?

c2) Beurteilen Sie die folgende Lösung:
Es wird die Oberklasse `Nutztier` von Henne und Kuh erstellt und in dieser die Methode `getAntibiotikaMenge()` erstellt.

Lösung:

1)

13P



2)

a)

12P

```
class Henne extends DruckObjekt{  
    private int antibiotikaMenge;  
  
    public int getAntibiotikaMenge() {  
        return antibiotikaMenge;  
    }  
  
    public void setAntibiotikaMenge(int pAntibiotikaMenge) {  
        antibiotikaMenge = pAntibiotikaMenge;  
    }  
  
    public Henne() {  
    }  
  
    public Henne(int pAntibiotikaMenge){  
        antibiotikaMenge = pAntibiotikaMenge;  
    }  
  
    public void printAll(){  
        System.out.println("Hennen-Antibiotikamenge="+antibiotikaMenge);  
    }  
}
```

6P

```
class Auto extends DruckObjekt{  
    private int maxSpeed;  
  
    public int getMaxSpeed() {  
        return maxSpeed;  
    }  
  
    public void setMaxSpeed(int pMaxSpeed) {  
        maxSpeed = pMaxSpeed;  
    }  
  
    public Auto() {  
    }  
  
    public Auto(int pMaxSpeed){  
        maxSpeed = pMaxSpeed;  
    }  
  
    public void printAll(){  
        System.out.println("Auto max. Geschw. =" + maxSpeed);  
    }  
}
```

6P

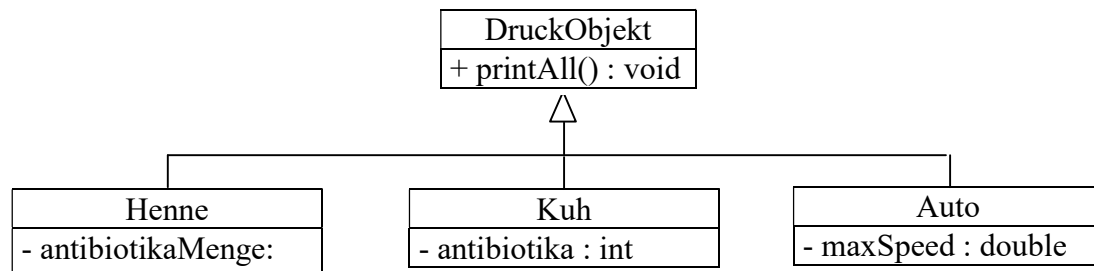
b)

b1)

Zeichnung:

6P

3P



```
abstract class DruckObjekt{
    public abstract void printAll();
}
```

3P

und jeweils bei den Unterklassen extends hinzufügen.

b2)

Die Methode printAll() abstract machen

2P

b3)

```
public static void main(String[] args) {
    int i;
    DruckObjekt[] druckObjekte = new DruckObjekt[3];
    Kuh mika = new Kuh(3);
    Henne berta = new Henne(4);
    Auto ente = new Auto(80);
    druckObjekte[0] = mika;
    druckObjekte[1] = new Henne(4);
    druckObjekte[2] = new Auto(80);
    for(i=0;i<3;i++){
        druckObjekte[i].printAll();
    }
}
```

9P

c1)

Dadurch erbt die Klasse Auto u.a. die Methode getAntibiotikaMenge().

4P

Dadurch könnte man die Antibiotikamenge eines Autos herausfinden, obwohl ein Auto kein Antibiotika gespritzt bekommt.

c2)

Dann hätte das Objekt Henne (und auch Kuh) zwei Oberklassen.

4P

Mehrfachvererbung ist aber in Java nicht erlaubt.

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

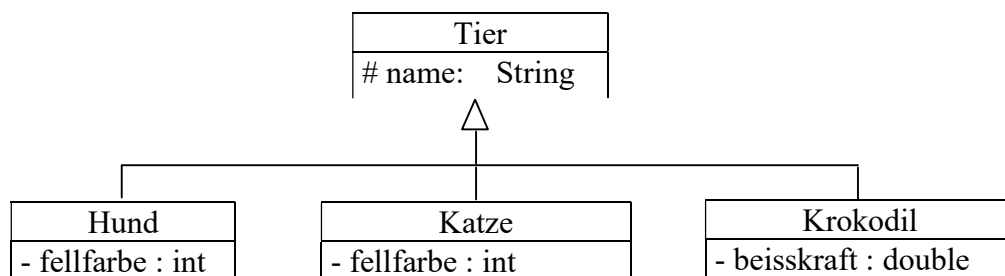
1) 15P
Eine natürliche Zahl n ($n \geq 2$) heißt Primzahl, wenn sie nur durch 1 und sich selbst teilbar ist.
Beispiele für Primzahlen: 2, 3, 5, 7, 11, 13,

Von einer ganzen Zahl $z \geq 2$ soll festgestellt werden, ob diese eine Primzahl ist.
Erstellen Sie dazu ein Struktogramm.

Hinweis zum Algorithmus:

Um von einer Zahl z festzustellen, ob sie eine Primzahl ist, versucht man, diese durch alle Zahlen 1, 2, 3, 4, ..., z zu teilen. Wenn die Anzahl der erfolgreichen (Zahl ist teilbar) Versuche gleich 2 ist, hat man eine Primzahl.

2) 50P
Gegeben ist das folgende abgespeckte, **fest vorgegebene** UML-Diagramm.
Die Klassenhierarchie darf also **nicht** verändert werden.



a) 39P
Erzeugen Sie aus dem folgenden UML-Diagramm die entsprechenden Klassen mit den jeweiligen Konstruktoren, set- und get-Methoden. Keine weiteren Attribute einfügen!
Die Konstruktoren müssen jeweils Parameter enthalten.
Wichtig: Jede Farbe wird durch einen Integer-Wert dargestellt, wie z.B:
... -100: weiß -99:gelb,
Umgekehrt entspricht jedem Integer-Wert eine Farbe!
Bitte keine "Umrechnungstabelle" der Zahlenwerte in tatsächliche Farben erstellen!
Hier ist eine Farbe ein Zahlenwert, mehr nicht!

b)

11P

Das Programm soll multipersonal entwickelt werden: Zu Testzwecken sollen die Entwickler der einzelnen Klassen gezwungen werden, die Methode `void getBeschreibung()` zu entwickeln, die die Namen der Attribute mit den zugehörigen Werten auf dem Bildschirm ausgibt.

In einem Feld sollen verschiedene Tiere (Hunde, Katzen, Krokodile) abgespeichert werden. Danach sollen die Werte der Attribute mit der Methode `getBeschreibung()` ausgegeben werden. Machen Sie folgendes in der Methode `main()`

b1) Erzeugen Sie einen Hund, eine Katze und ein Krokodil.

b2) Speichern Sie diese 3 Tiere in dem Feld.

b3) Geben Sie mit Hilfe einer Schleife und der Methode `getBeschreibung()` die Eigenschaften der jeweiligen Tiere auf dem Bildschirm aus.

Lösung:

public class MainKlassenarbeit {		11P
public static void main(String[] args) {		
int i;		
Katze katze;		
Hund hund;		
Krokodil krokodil;		
Tier tiere[];	1P	
tiere = new Tier[3];	1P	
katze = new Katze("Ute", 2);	1P	
tiere[0] = katze;	1P	
hund = new Hund("Rex", 3);	1P	
tiere[1] = hund;	1P	
krokodil = new Krokodil("Krok", 30);	1P	
tiere[2] = krokodil;	1P	
System.out.println("Beschreibung der Tiere:");		
for(i=0; i<tiere.length; i++){	3P	
System.out.println(tiere[i].getBeschreibung());		
}		
}		
}		
 abstract class Tier{	1P	11P
protected String name;	1P	
 public Tier(String pName){	2P	
name = pName;		
}		
 public void setName(String pName){	2P	
name=pName;		
}		
 public String getName(){	2P	
return(name);		
}		
 abstract public String getBeschreibung();	3P	
}		
 class Katze extends Tier {		10P
int fellfarbe;		
 public Katze(String pName, int pFellfarbe){		
super(pName);		
fellfarbe = pFellfarbe;		
}		
 public String getBeschreibung(){		
String s;		
s="Name="+name+" Fellfarbe="+fellfarbe;		
return s;		
}		
 public void setFellfarbe(int pFellfarbe){		
fellfarbe = pFellfarbe;		
}		

```

    public int getFellfarbe() {
        return fellfarbe;
    }
}

```

```

class Hund extends Tier {
    private int fellfarbe;

```

10P

```

    public Hund(String pName, int pFellfarbe) {
        super(pName);
        fellfarbe = pFellfarbe;
    }

```

```

    public String getBeschreibung() {
        String s;
        s="Name="+name+" Fellfarbe="+fellfarbe;
        return s;
    }

```

```

    public void setFellfarbe(int pFellfarbe) {
        fellfarbe = pFellfarbe;
    }

```

```

    public int getFellfarbe() {
        return fellfarbe;
    }
}

```

```

class Krokodil extends Tier{
    private double beisskraft;

```

10P

```

    Krokodil(String pName, double pBeisskreaft){
        super(pName);
        beisskraft = pBeisskreaft;
    }

```

```

    public String getBeschreibung() {
        String s;
        s="Name="+name+" Beisskraft="+beisskraft;
        return s;
    }

```

```

    public void setBeisskraft(double pBeisskraft) {
        beisskraft = pBeisskraft;
    }

```

```

    public double getBeisskraft() {
        return beisskraft;
    }
}

```

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

I)

Auf einem Bauernhof gibt es Hennen und Kühe (siehe unten)

Die Klasse Bauernhof soll folgende Klassen-Methoden (mit static) enthalten

.... erzeugeNutztier()

Diese gibt abhängig vom Zufall entweder eine Kuh oder eine Henne zurück.

Legeleistung bzw. Milchleistung soll eine Zufallszahl zwischen 0 und 10 sein.

void printNutztiere(...)

Diese gibt die Attribute einer Menge von Kühen und Hennen auf dem Bildschirm aus.

1)

35P

Erstellen Sie die Klasse Bauernhof mit 0 Attributen und genau diesen 2 Methoden.

2)

15P

a) In der Methode main sollen in einem Feld 3 Nutztiere (Kühe oder Hennen) angelegt werden.

Realisieren Sie dies mit Hilfe der Methode erzeugeNutztier()

b) Geben Sie danach mit Hilfe der Methode printNutztiere(...) die Attribute dieser 3 Tiere auf dem Bildschirm aus.

Bemerkungen:

1) Beispiel für die Erzeugung einer Zufallszahl:

```
public static void main(String[] args) {
    double zufall1, zufall2;
    // liefert eine double-Zahl zwischen 0 und 1
    zufall1 = Math.random();
    if (zufall1 < 0.5) {
        zufall2 = Math.random() * 10;
    }
    else {
        zufall2 = (int) (Math.random() * 10);
    }
}
```

2) Diese Klassen und Methoden wurden schon implementiert. Auf diese darf man sich beziehen, bzw. diese dürfen verwendet werden.

```
class Henne extends Nutztier{
    private int legeleistung;;

    public Henne(int legeleistung) {
        this.legeleistung = legeleistung;
    }

    public int getLegeleistung() {
        return legeleistung;
    }

    public void setLegeleistung(int legeleistung) {
        this.legeleistung = legeleistung;
    }

    public void prinAllAttributes(){
        System.out.println("Legeleistung = "+legeleistung);
    }
}

class Kuh extends Nutztier{
    private double milchleistung;

    public Kuh(double milchleistung) {
        this.milchleistung = milchleistung;
    }

    public double getMilchleistung() {
        return milchleistung;
    }

    public void setMilchleistung(int milchleistung) {
        this.milchleistung = milchleistung;
    }

    public void prinAllAttributes(){
        System.out.println("Milchleistung = "+milchleistung);
    }
}

abstract class Nutztier{
    abstract public void prinAllAttributes();
}
```


Lösung:

```
package e2fi_19_05_12_nachtermin2;

public class MainE2fi_19_05_12_nachtermin2 {
    public static void main(String[] args) {
        Nutztier[] nutztiere = new Nutztier[3];
        nutztiere[0]=Bauernhof.erzeugeNutztier();
        nutztiere[1]=Bauernhof.erzeugeNutztier();
        nutztiere[2]=Bauernhof.erzeugeNutztier();

        Bauernhof.printNutztiere(nutztiere);
    }
}

class Bauernhof{
    public static Nutztier erzeugeNutztier(){
        double zufall;
        Nutztier nutztier;
        zufall = Math.random();
        if (zufall<0.5){
            nutztier = new Kuh(Math.random()*10);
        }
        else{
            nutztier = new Henne((int) (Math.random()*10));
        }
        return nutztier;
    }

    public static void printNutztiere(Nutztier[] nutztiere){
        int i;
        for(i=0;i<nutztiere.length;i++){
            nutztiere[i].prinAllAttributes();
        }
    }
}

class Henne extends Nutztier{
    private int legeleistung;;

    public Henne(int legeleistung) {
        this.legeleistung = legeleistung;
    }

    public int getLegeleistung() {
        return legeleistung;
    }

    public void setLegeleistung(int legeleistung) {
        this.legeleistung = legeleistung;
    }

    public void prinAllAttributes(){
        System.out.println("Legeleistung = "+legeleistung);
    }
}
```

```
class Kuh extends Nutztier{
    private double milchleistung;

    public Kuh(double milchleistung) {
        this.milchleistung = milchleistung;
    }

    public double getMilchleistung() {
        return milchleistung;
    }

    public void setMilchleistung(int milchleistung) {
        this.milchleistung = milchleistung;
    }

    public void prinAllAttributes(){
        System.out.println("Milchleistung = "+milchleistung);
    }
}

abstract class Nutztier{
    abstract public void prinAllAttributes();
}
```