Vererbung

Beispielklasse `Person`:

```
public abstract class Person {

	private String addresse;
	private String telefonnummer;

	public Person(String addresse) {
		this.addresse = addresse;
	}

	public void setTelefonnummer(String telefonnummer) {
		this.telefonnummer = telefonnummer;
	}

	public String getTelefonnummer() {
		return telefonnummer;
	}

	public abstract String getName();

}
```

Vererbung: Klasse `NatuerlichePerson` erbt von Klasse `Person`:

```
public class NatuerlichePerson extends Person {

	private String vname, nname;

	public NatuerlichePerson(String vname, String nname, String adr) {
		super(adr);
		this.vname = vname;
		this.nname = nname;
	}

	public String getName() {
		return vname+" "+nname;
	}

}
```

Verwendung:

```
public class Test2 {

	public static void main(String[] args) {
		Person p = new NatuerlichePerson("Max", "Muster",
                                         "1080 Wien; Alserstr. 17");
		p.setTelefonnummer("01/1234567");
		System.out.println(p.getName());
		System.out.println(p.getTelefonnummer());
	}
}
```

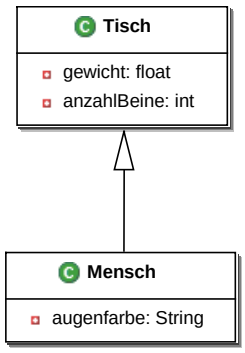
Bsp 1: Juristische Person

Ergänzen Sie das Beispiel der Übersichtsseite (1. Seite) um eine Juristische Person. Eine Juristische Person ist eine Person, die zusätzlich einen Firmennamen und eine Gesellschaftsform besitzt. Stellen Sie sinnvolle Methoden zur Verfügung, um mit den Daten arbeiten zu können.

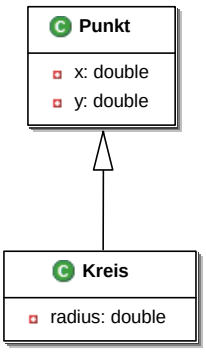
Bsp 2: Vererbung

Beurteilen Sie die Vererbungsbeziehung der folgenden Beispiele. Erklären Sie Kriterien einer sinnvoll eingesetzten Vererbung.

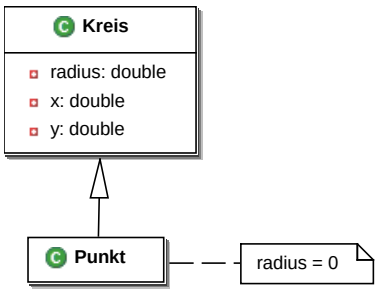
a)



b)



c)



Bsp 3: Gehaltsberechnung

Das Jahresgehalt eines normalen Angestellten berechnet sich über 14 * Monatsgehalt. Ein Manager kann zusätzlich zu seinem Grundgehalt noch eine Erfolgsprämie erhalten. Bilden Sie diesen Zusammenhang in Klassen mit einer entsprechenden Vererbungsbeziehung ab.

Bsp 4: Vererbungshierarchie

a) Bringen Sie die folgenden Begriffe in eine Vererbungshierarchie: Säugetier, Fisch, Mensch, Affe, Karpfen, Hecht, Forelle, Wal, Lebewesen, Kamel, Mann, Frau, Buch, Tier
b) Finden Sie zu jeder Klasse zumindest ein Datenelement.
c) Gibt es in Ihrer Hierarchie Klassen, welche sinnvoller Weise als abstrakt deklariert werden sollten?
d) Implementieren Sie zumindest 5 Klassen der Hierarchie in Java.

Bsp 5: Orchester

Ein Dirigent leitet mehrere Musikanten. Ein Musikant kann ein Sänger oder ein Instrumentalist sein. Ein Sänger ist durch dessen Name und den zu singenden Text spezifiziert. Ein Instrumentalist hat einen Namen, ein Instrument und (der Einfachheit halber) einen einzigen Ton, der mit dem Instrument gespielt werden kann.
Der Dirigent soll eingebbar oft einen Musikanten zufällig auswählen und zum Spielen auffordern, der daraufhin entweder seinen Text singt oder mit seinem Instrument den Ton spielt.

```
Klaus mit Saxophon: Quiiiietsch
Sandra mit Trommel: Bada-Bumm
Tom mit Gitarre: Klimper
Julia singt: La-La
Sandra mit Trommel: Bada-Bumm
Julia singt: La-La
Tom mit Gitarre: Klimper
Klaus mit Saxophon: Quiiiietsch
...
```

Bsp 6: Erweitertes Orchester

Erweitern Sie das Orchester aus Beispiel 4 um einen Stepptänzer, der ebenfalls vom Dirigenten geleitet wird. Muss durch die Einführung des Stepptänzers an der Klasse Dirigent etwas verändert werden?

Bsp 7: Grafische Objekte

Implementieren Sie folgende Klassen und finden Sie eine sinnvolle Vererbungshierarchie: `GrafischesObjekt`, `Punkt`, `Linie`, `Kreis`, `Rechteck`

a) Schreiben Sie Methoden für die Berechnung der Fläche der Figuren.
b) Schreiben Sie Methoden, um die Figuren in der Fläche zu verschieben (Translation).
c) Schreiben Sie eine Klasse `Graph`, die eine Menge von grafischen Objekten halten kann. Implementieren Sie weiters eine Methode `selektiere`, an die ein `Rechteck` übergeben wird. Die Methode soll ein Array von allen grafischen Objekten innerhalb des zu selektierenden Rechtecks zurückliefern. Erweitern Sie dazu die grafischen Objekte entsprechend.

Bsp 8: Array-Filter

Schreiben Sie eine Klasse `ArrayFilter`, bei der dynamisch mehrere Filteralgorithmen registrierbar sind. Implementieren Sie weiters eine Methode `filter`, die ein Integer-Array als Parameter besitzt und als Rückgabewert ein Integer-Array zurückgibt. Die Methode soll alle registrierten Filteralgorithmen auf das übergebene Array anwenden und das so entstehende Array zurückliefern.
Implementieren Sie zumindest folgende Filteralgorithmen:

- `MinFilter(int untergrenze)`: Filtert alle Elemente unterhalb des Grenzwerts aus.
- `MaxFilter(int obergrenze)`: Filtert alle Elemente oberhalb des Grenzwerts aus.
- `ElementFilter(int[] erlaubteElemente)`: Filtert alle Elemente, die nicht in der Menge der erlaubten Elemente sind, aus.

Folgender Testcode zeigt die Verwendung des Array-Filters:

```
ArrayFilter arrayFilter = new ArrayFilter();
arrayFilter.register(new MinFilter(10));
arrayFilter.register(new MaxFilter(50));
int[] erlaubteElemente = { 1, 10, 20, 30, 40, 50, 60 };
arrayFilter.register(new ElementFilter(erlaubteElemente));
int[] testArray1 = { 1, 10, -15, 20, 13, 17, 60, 50 };
int[] filteredTestArray1 = arrayFilter.filter(testArray1);
// filteredTestArray1 sollte sein: { 10, 20, 50 }
```