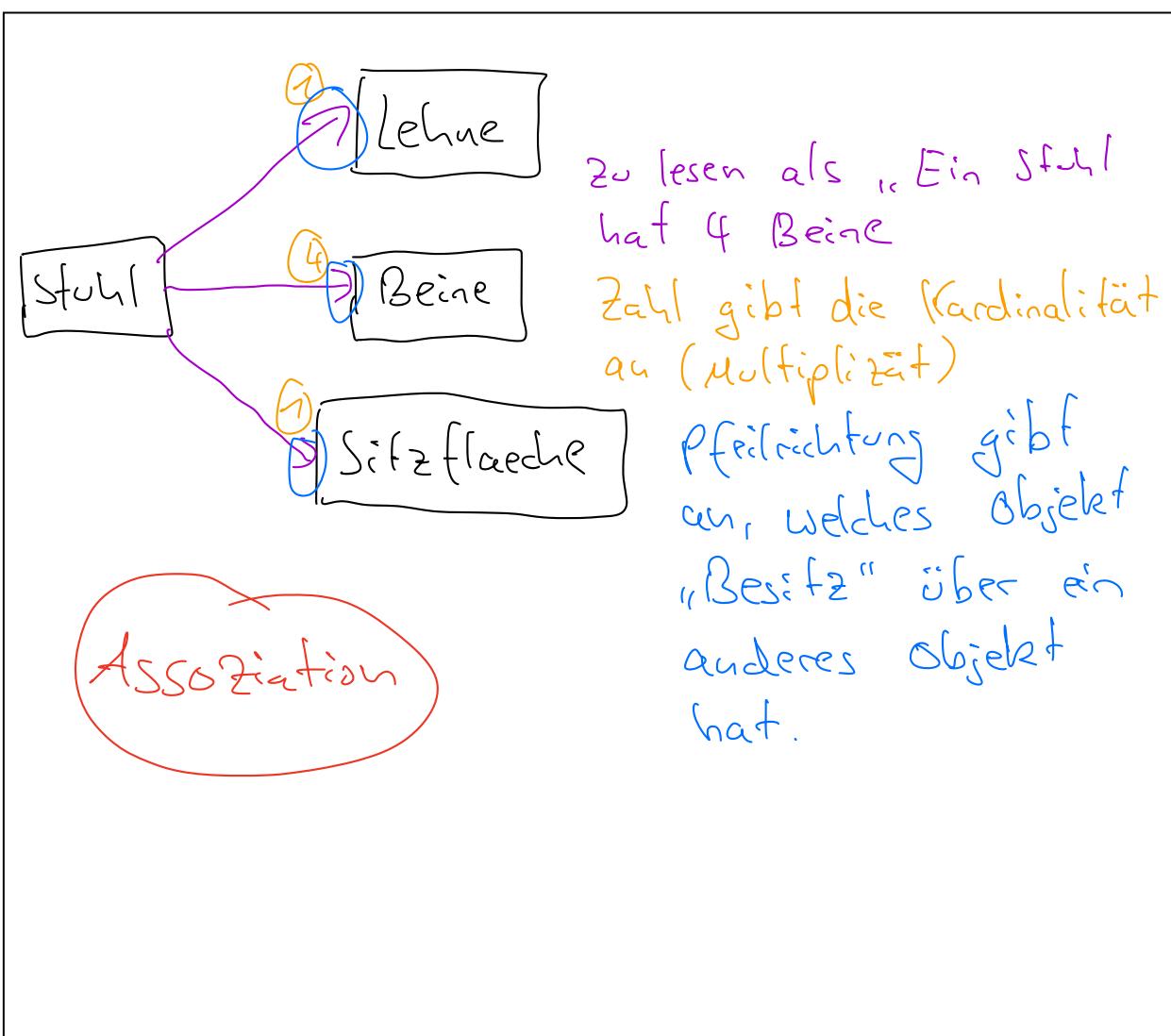




## Vererbung



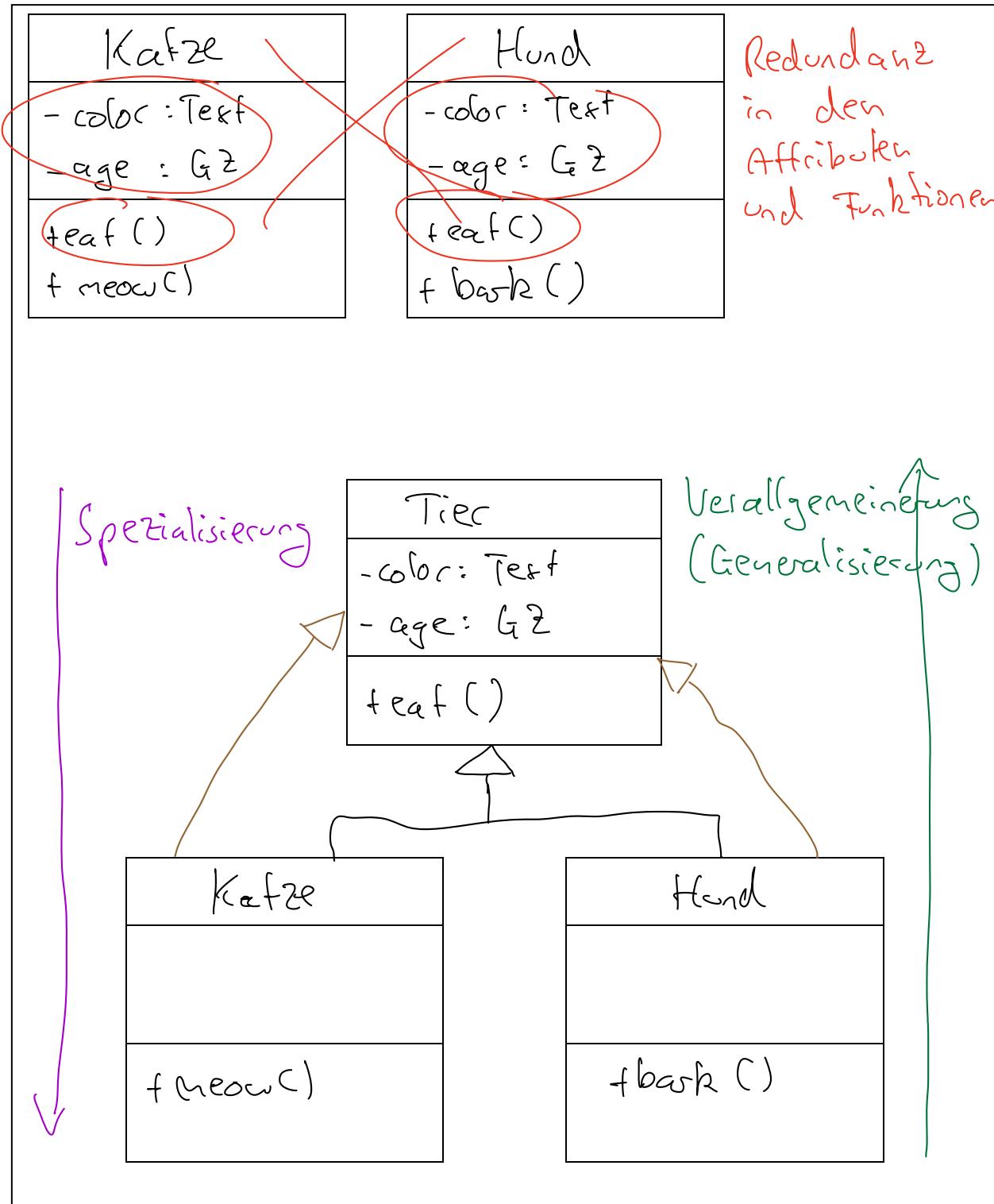
HAS-A Beziehungen zwischen Objekten → Assoziationen





## IS-A Beziehungen zwischen Objekte → Vererbung

Beispiel: Eine Anwendung soll Hunde und Katze abbilden



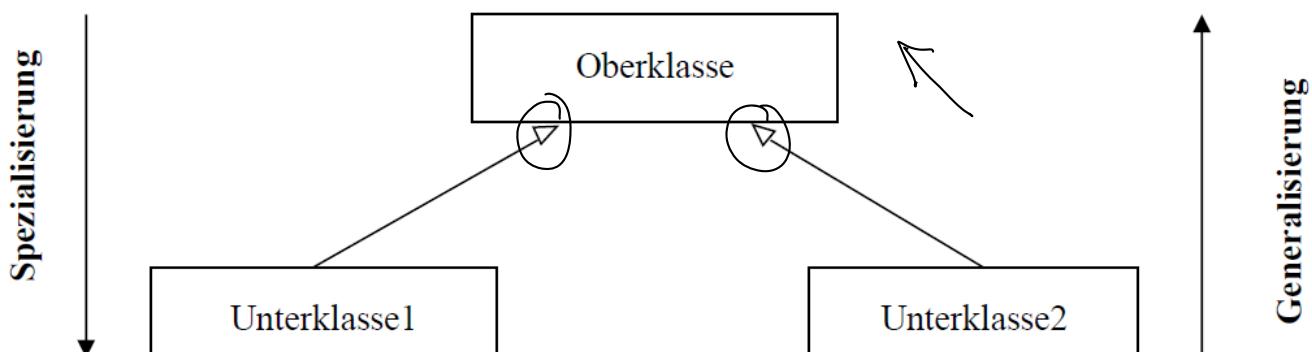
Aus dem UML-Klassendiagramm lassen sich folgende Informationen ableiten:



Die **Vererbung** (inheritance) beschreibt Beziehungen zwischen einer allgemeinen Klasse (**Basisklasse**, **Oberklasse**) und spezialisierten Klassen (**Unterklassen**) und hat folgende Eigenschaften:

- die Unterklassie verfügt über alle Informationen der Oberklassie (Attribute, Operationen und Beziehungen werden vererbt)
  - jedes Objekt einer Unterklassie ist auch Objekt der Oberklassie
  - die Unterklassie kann das durch die Oberklassie definierte Verhalten verfeinern, erweitern und/oder einschränken (durch zusätzliche Attribute, Operationen, Beziehungen und/oder Überschreiben von bestehenden Operationen)

## UML-Darstellung:





Jede Unterklass kann selbst Oberklasse einer oder mehrerer Unterklassen sein. So entsteht eine evtl. komplexe Baumstruktur (**Klassenhierarchie**).

Bei der **Einfachvererbung** besitzt jede Unterklass genau eine direkte Oberklasse. Bei der **Mehrfachvererbung** kann eine Unterklass mehrere direkte Oberklassen besitzen. Es entsteht eine Netzstruktur. Die Mehrfachvererbung sollte möglichst vermieden werden, da Konflikte auftreten können (z. B. wenn Attribute oder Operationen mit demselben Namen von beiden Oberklassen geerbt werden sollten). Java unterstützt nur die Einfachvererbung.

Eine bestimmte Operation einer Oberklasse kann in einer Unterklass neu definiert werden (**Überschreiben, Overriding**), dabei müssen Anzahl und Typen der Parameter der Operation (Signatur) übereinstimmen. In der "neuen" Version der Operation kann die "ursprüngliche" Operation aufgerufen werden.

Wichtig: Das Überschreiben darf nicht mit dem Überladen von Operationen verwechselt werden.

### Polymorphie

Polymorphie-Prinzip: Ein Operationaufruf kann unterschiedliche Auswirkungen haben.

Es gibt zwei Arten von Polymorphie:

- **statische Polymorphie:** mehrere Operationen haben denselben Namen, unterscheiden sich jedoch in Anzahl und/oder Datentyp der Parameter (**Signatur**). In diesem Fall spricht man von Überladen (overloading). Es ist schon während der Übersetzung des Programms möglich, anhand der aktuellen Parameterliste eines Operationaufrufs die passende Operation zu finden (daher statisch).
- **dynamische Polymorphie:** dieselbe Operation kann unterschiedliche Reaktionen auslösen, je nach Typ des Objekts (da die Operation in der Klassenhierarchie evtl. überschrieben wird). Z.B. wird beim Aufruf einer Operation zeichnen() eine geometrische Figur anders gezeichnet, je nach Typ der Figur (Kreis und Rechteck sind Unterklassen der Klasse GeomFigur mit jeweils einer eigenen Version von zeichnen() ). Die Bestimmung der relevanten Operation kann erst zur Laufzeit erfolgen (spätes Binden), wenn klar ist, um welches Objekt es sich in dem Moment wirklich handelt.

Der Begriff Polymorphie ohne Zusatz steht für dynamische Polymorphie.

### Vererbung von Attributen

Private Attribute einer Oberklasse sind nur innerhalb der Oberklasse sichtbar, die Unterklass erbt sie zwar, kann sie aber nicht direkt lesen / beschreiben. Objekte der Unterklassen können also nur per Operationaufrufe auf private Attribute zugreifen.

Attribute einer Oberklasse, die jedoch als **protected** (#) definiert werden, sind sowohl in dieser Oberklasse als auch in allen Unterklassen sichtbar (d.h. direkter Zugriff möglich). Dadurch wird allerdings das Datenkapselungsprinzip verletzt, wenn z.B. von einem Objekt der Unterkasse aus Werte von Attributen, die aus der Oberklasse stammen, direkt also unkontrolliert (nicht über Eine Operation) verändert werden.



## Übungsaufgaben

### Aufgabe 1:

- a) Generalisieren Sie: **Kugel, Zylinder, Würfel**

→ Oberklasse1: 3-Dim-Figur

Generalisieren Sie: **Dreieck, Quadrat, Kreis**

→ Oberklasse2: 2-Dim-Figur

- b) Generalisieren Sie: Oberklasse1 und Oberklasse2

Figur

- c) Stellen Sie die Klassenhierarchie von a) und b) in einem Klassendiagramm dar (ohne Attribute und ohne Operationen).

### Aufgabe 2:

- a) Generalisieren Sie: **Limonade, Fruchtsaft**

→ Oberklasse1: \_\_\_\_\_

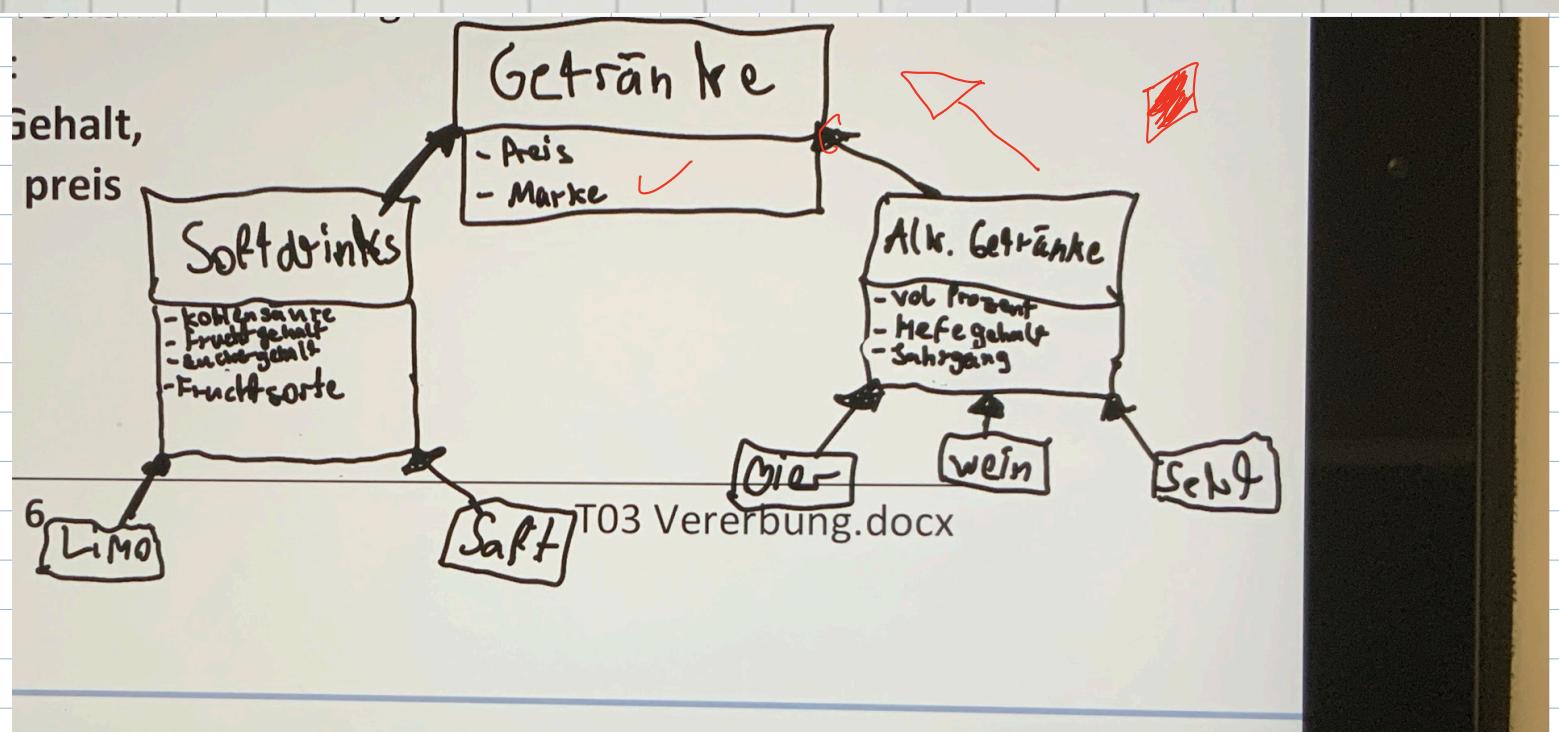
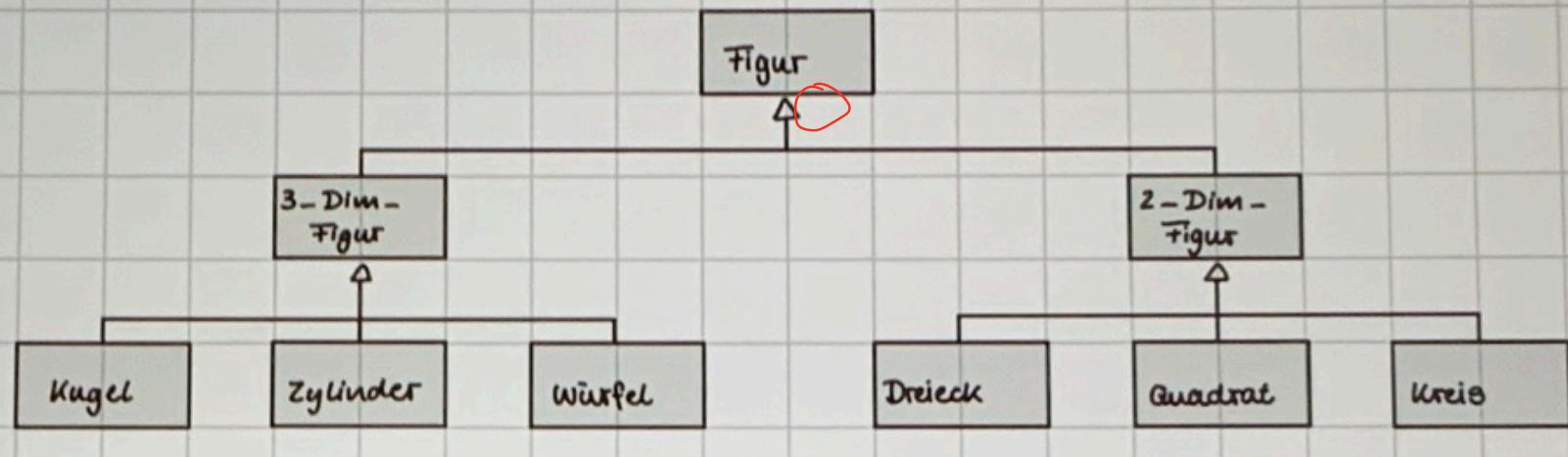
Generalisieren Sie: **Bier, Wein, Sekt**

→ Oberklasse2: \_\_\_\_\_

- b) Generalisieren Sie: Oberklasse1 und Oberklasse2

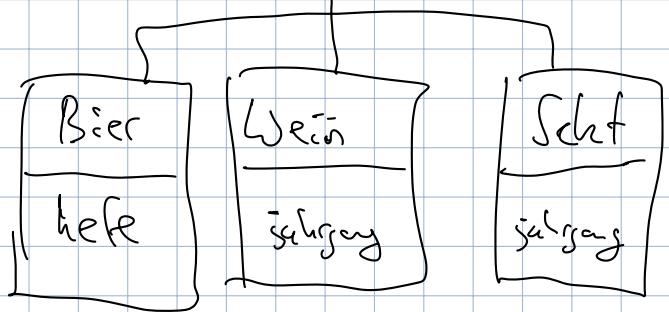
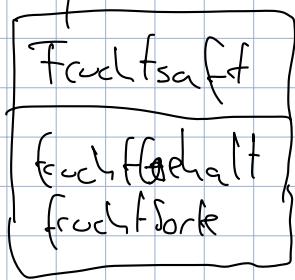
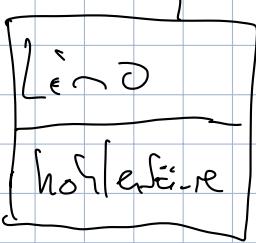
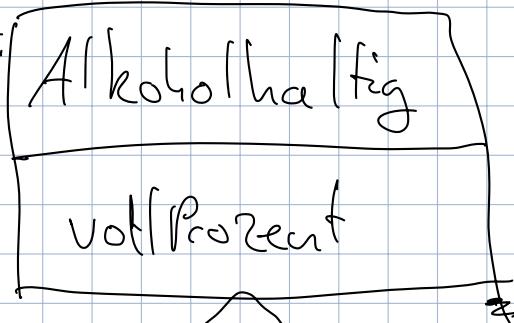
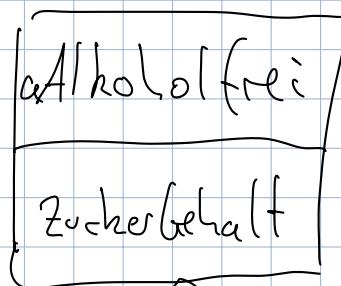
\_\_\_\_\_

- c) Stellen Sie die Klassenhierarchie von a) und b) in einem Klassendiagramm dar. Tragen Sie dabei folgende Attribute in die zugehörige Klasse ein:  
**kohlensäure, fruchtgehalt, volProzent, hefeGehalt, zuckerGehalt, marke, fruchtSorte, jahrgang, preis**





KohlenSäure





## Übungsaufgaben

### Aufgabe 3:

In der Schule gibt es Schüler und Lehrer, für die Name und Geburtsjahr gespeichert werden. Schüler haben außerdem 10 Noten und Lehrer 2 Unterrichtsfächer. Es soll möglich sein, das Alter und bei Schülern zusätzlich den erreichten Durchschnitt zu berechnen. Überlegen Sie sich eine Oberklasse den geeigneten Attributen und Operationen.

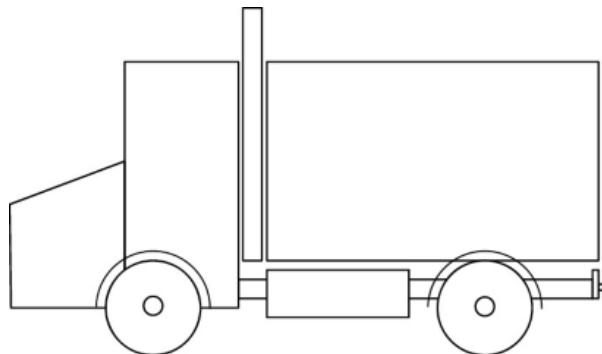
Skizzieren Sie das UML-Klassendiagramm.

### Aufgabe 4:

Ein Softwareunternehmen entwickelt ein System zur Verwaltung von Fahrzeugen. Es gibt verschiedene Arten von Fahrzeugen, wie Autos und Lastwagen, die alle gemeinsame Eigenschaften und Methoden haben, aber auch spezifische Merkmale aufweisen.

In dem System gibt es eine allgemeine Klasse Fahrzeug, die die grundlegenden Eigenschaften und Methoden aller Fahrzeuge definiert. Jedes Fahrzeug hat eine Fahrgestellnummer, eine Marke und ein Baujahr. Außerdem kann jedes Fahrzeug starten und stoppen, was durch die Methoden `starten()` und `stoppen()` dargestellt wird.

Von der Klasse Fahrzeug erben zwei spezialisierte Klassen: Auto und Lastwagen. Die Klasse Auto hat zusätzlich die Eigenschaft `anzahlSitze`, die die Anzahl der Sitze im Auto angibt. Die Klasse Lastwagen hat die zusätzliche Eigenschaft `ladeKapazität`, die die maximale Ladekapazität des Lastwagens in Tonnen angibt.



- Lesen Sie den Text durch.
- Erstellen Sie ein UML-Klassendiagramm, das die Klassen Fahrzeug, Auto und Lastwagen sowie deren Beziehungen darstellt. Überlegen Sie sich weitere Operationen für die Unterklassen.

