Einführung in die Objektorientierung (OO)

I) Warum OO?

II) Grundbegriffe der OO

III) Darstellung von Klassen und Objekten

IV) Kapselung

I) Warum OO?

1) Früher: Prozedurale / strukturierte Programmierung

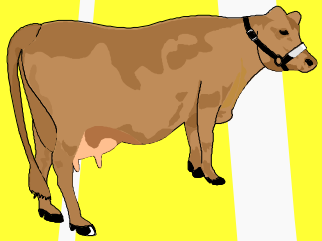
- Funktionsorientierte Zerlegung

  (Aufgabe wird zerlegt in Programme)

- Trennung von Daten und Funktionen

  (Nachteil: jede Funktion definiert „eigene Daten“)

Beispiel: Kuh „elsa“



füttern

↓

auf die Weide treiben

↓

von der Weide zurücktreiben

↓

melken

4 unabhängige Programme

Merke: Mögliche Kandidaten für Funktionen sind die Verben der Aufgabenstellung

2) OOP: Objektorientierte Programmierung

- Datenorientierte Zerlegung
- Funktionen werden den Daten im Objekt zugeordnet

Beispiel: Kuh „elsa“

	Kuh: „elsa“
Attribute (Daten)	gewicht: 800 kg alter: 5 Jahre milchbestand: 25 ltr
Operationen (Methoden)	melken(): füttern():

Merke: Mögliche Kandidaten für Klassen sind die Substantive der Aufgabenstellung

3) Softwarequalität

Externe Qualitätsfaktoren (vom Benutzer bemerkbar)

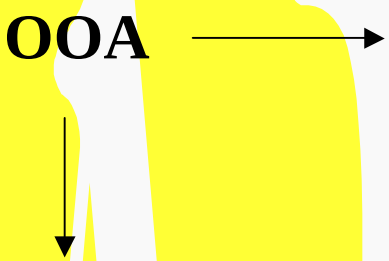
- korrekt
- robust
- erweiterbar
- wiederverwendbar
- verträglich
- effektiv
- übertragbar (portabel)
- einfach (leicht benutzbar)
- zweckmäßig
- pünktlich
- nachweisbar
- integer
- reparabel
- wirtschaftlich

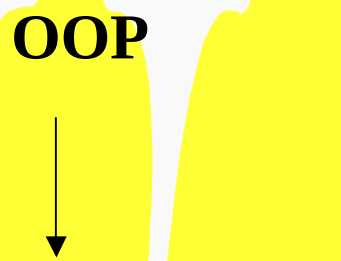
Interne Qualitätsfaktoren

(nur vom Spezialisten bemerkbar)

- modular, strukturiert, ...

4) Begriffsabgrenzung: OOA/OOD/OOP





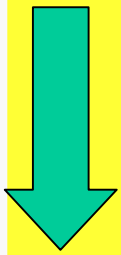
Modell der Systemanforder-ungen

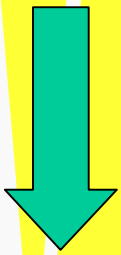
EDV-unabh. Beschreibung von Details und Randbedingungen

Implementierung (EDV-spezifisch)

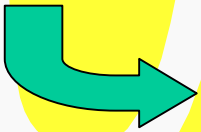
„Anwendersicht“ **„Entwicklersicht“** **„Entwicklersicht“**

(Erst das WAS und dann das WIE)





Beschreibung aller 3 Phasen mit UML (Unified Modeling Language)



Dokumentation und Visualisierung

II) Grundbegriffe der OO

1) Abstraktion: Reduktion der komplexen Welt auf das Wesentliche

z.B. Reduktion einer Person auf Name und Adresse

2) Objekt = Abstraktes Element der realen Welt

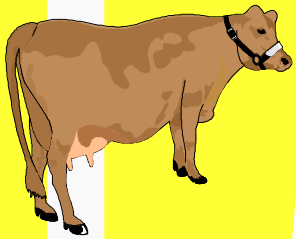
- Abbildung realer Dinge

  (Bilder, Pläne)

- Abbildung von Prozessen

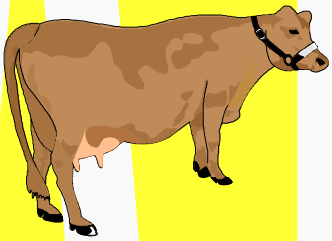
  (z.B. Telefongespräch)

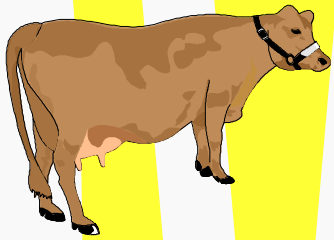
Beispiel: Kuh „elsa“



Objekteigenschaften

- **Zustand: Daten (Attribute)**
- **Verhalten: Methoden (Operationen)**
- **Identität: Objekt-Id (eindeutige Ident.-Nr)**





Kuh „elsa“ **Kuh „britta“**

2 Objekte sind auch dann verschieden, wenn sie sich im gleichen Zustand befinden !!!

Festlegung: Objektnamen als Substantiv im Singular mit kleinem Anfangsbuchstaben

z. B.: eineKuh, einAuto, einePerson

3) Klasse

Klasse = Beschreibung einer Menge von ähnlichen Objekten

Beispiel: Klasse „Auto“

R19 mit BB-UY 331 ist Element (Exemplar, Instanz, Objekt) dieser Klasse!

Die Klasse hängt immer von der gestellten Aufgabe ab!

- Autoverkauf → Klasse „Auto“ mit Ausstattung, Lackierung,...

- Fahrschule → Klasse „Auto“ mit Drehzahl, Geschwindigkeit,...

Festlegung: Klassennamen als Substantiv im Singular mit großem Anfangsbuchstaben

z. B.: Auto, Person, ...

	Klasse	Objekt
a)	statisch ↳ zur Übersetzungs-zeit ↳ Stack (Kellerspeicher)	dynamisch ↳ zur Laufzeit ↳ Heap (Haldenspeicher)
b)	kennt ihre Objekte nicht	kennt die zugehörige Klasse
c)	kann auch ohne Objekte existieren	kann nur als Element einer Klasse existieren
d)	legt Strukturen und Verhalten aller zuge-hörigen Objekte fest ↳ Vorlage (Stempel)	hat das spezifische Verhalten der Klasse

III) Darstellung von Klassen und Objekten in UML

3 Rubriken

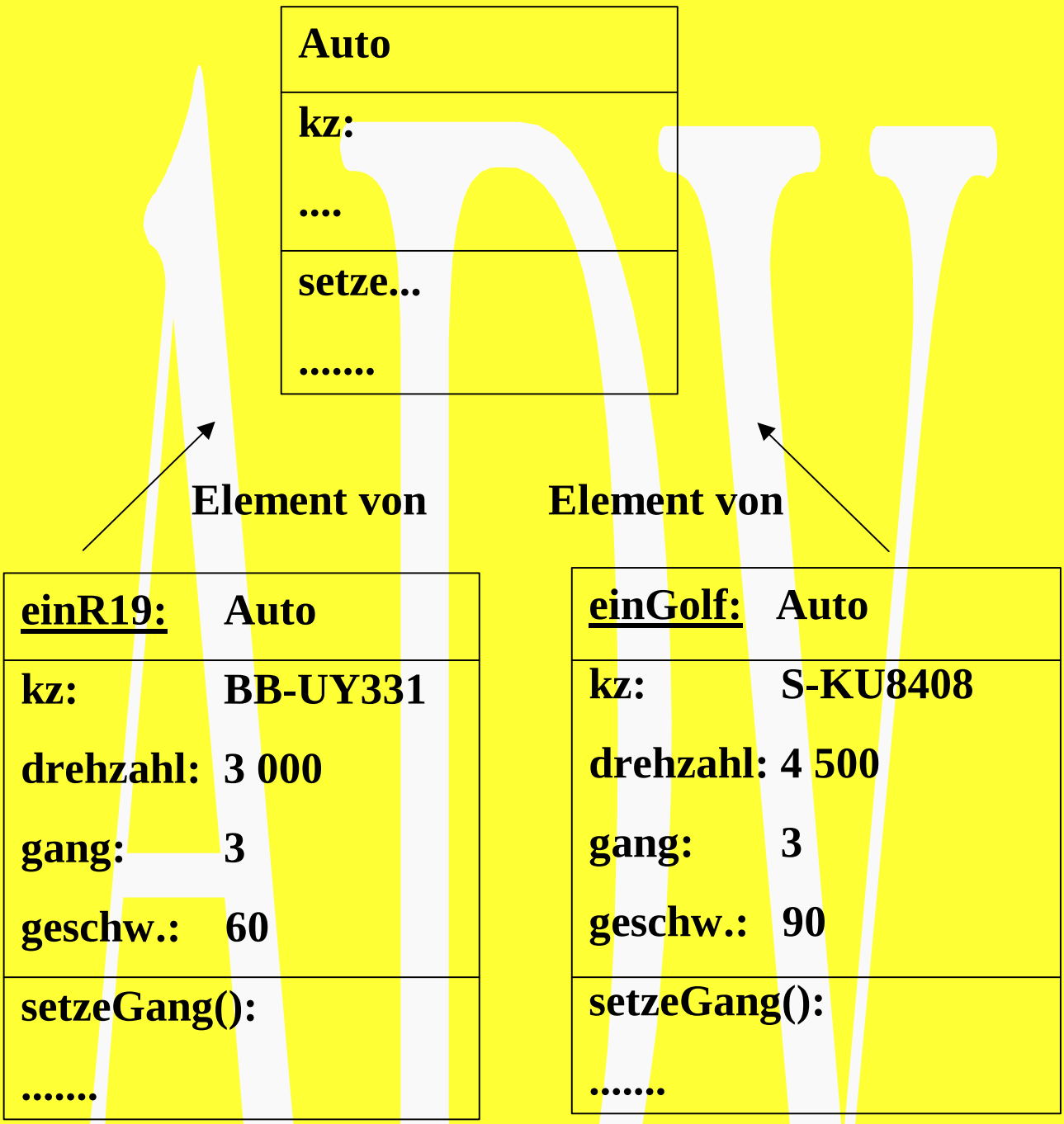
Klassenname	
Attribute	Zustand
Operationen	Verhalten

Beispiel: Die Klasse „Auto“

Auto	
kz:	zeichenkette
drehzahl:	ganzzahl
gang:	ganzzahl
geschw.:	reellzahl
setzeGang(ganzzahl):	keine Rückgabe
setzeGeschw.(reellzahl):	keine Rückgabe
gibGang(keine Übergabe):	ganzzahl
gibGeschw.(keine Übergabe):	reellzahl

Daten-typen festlegen

Signatu-ren festlegen



Festlegung: Attribut- und Operationennamen mit kleinem Anfangsbuchstaben

z. B.: gang, setzeGang() ...

Botschaften (Nachrichten)

Wie kann die Operation „setzeGang“ für das Objekt „einR19“ durchgeführt werden?

Dazu wird dem Objekt eine „Botschaft“ geschickt.

Diese aktiviert eine Operation gleichen Namens.

Im Gegensatz zur prozeduralen Programmierung sind in der OO die Operationen an Objekte gebunden.

objekt.botschaft(argument);

z. B.:

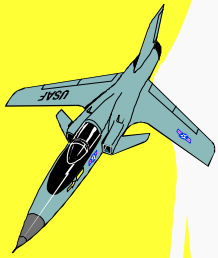
einR19.setzeGang(3);

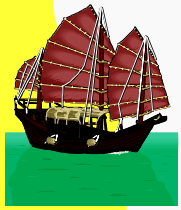
Anders als bei einer prozeduralen Lösung lassen sich in einer objektorientierten Lösung die Operationen nur über das Objekt ansprechen.
Das Objekt „einR19“ ist der Empfänger der Botschaft „setzeGang(3)“!
Das Objekt, das den Operationsaufruf enthält, ist der Sender der Botschaft.

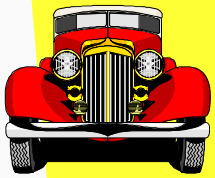
Objektattribute sind gekapselt und nach außen nur über entsprechende Operationen zugänglich.

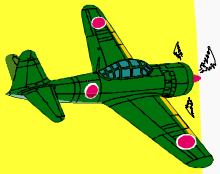
Eine Botschaft kann von einem Objekt nur interpretiert werden, wenn es eine dazu *passende* Operation besitzt.

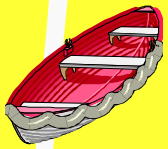
Wieviele Klassen sehen Sie ?

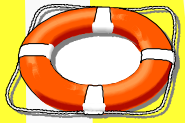


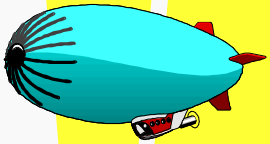


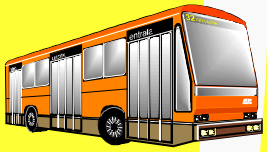








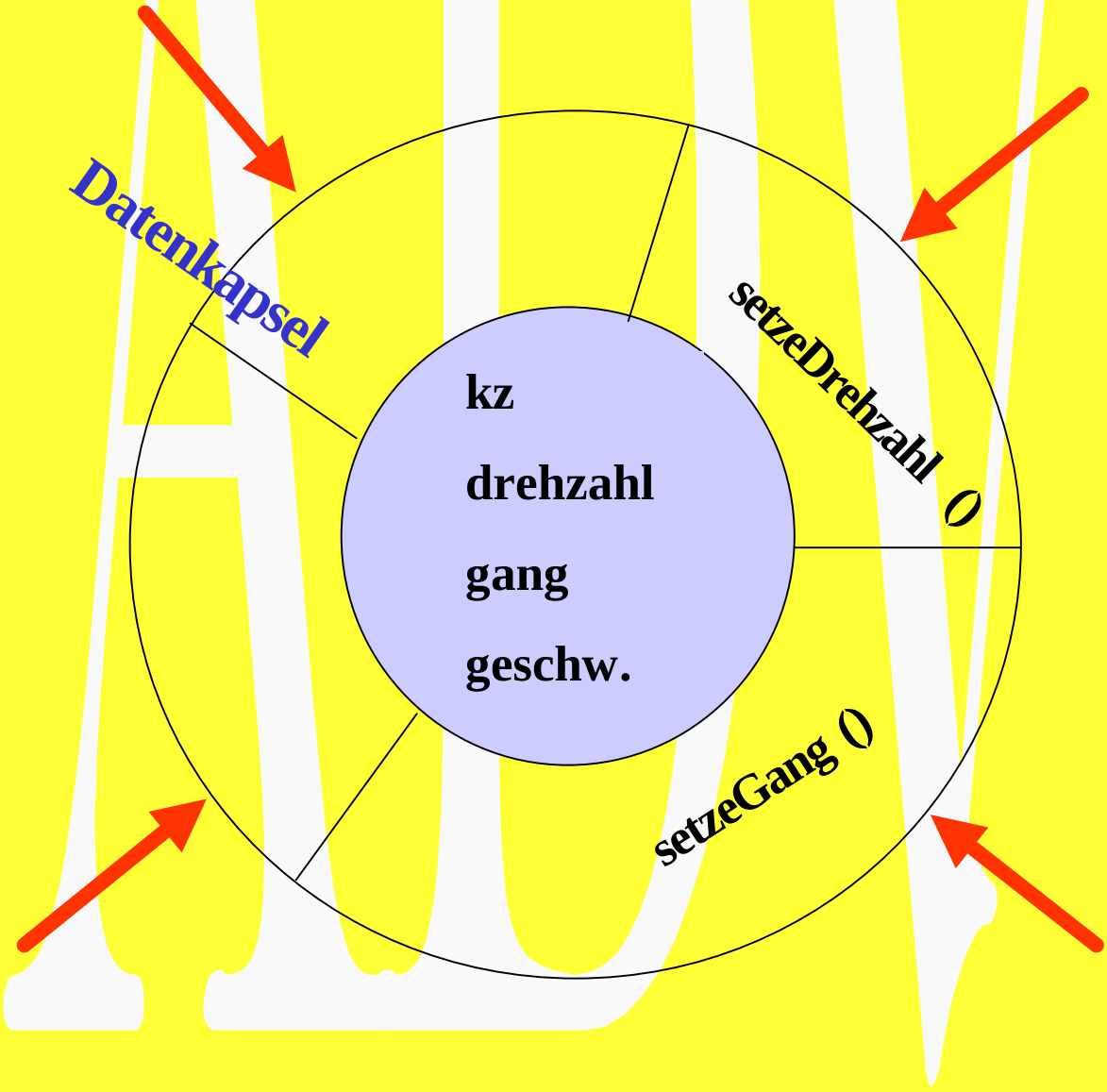




Brotz - Einführung in die OOP

IV) Kapselung

a) Was ist Kapselung?



Unter Kapselung versteht man die konzeptionelle Trennung der

Signatur einer Operation von deren Implementierung

beschreibt die Benutzung **konkrete Realisierung**

→ **Black-Box, Information Hiding, Geheimnisprinzip!**

Für den Benutzer einer Operation ist nur die Handhabung der Operation wichtig, nicht wie die Operation realisiert ist!

Die Operationen legen sich schützend um die Daten

→ **Datenkapsel**

Attributwerte dürfen nicht direkt verändert werden, sondern nur über entsprechende Operationen (Schnittstelle)!

Das Objekt bietet Operationen an, um Werte auszulesen.

b) Vor- u. Nachteile der Kapselung

+ Änderungen der Implementierung wirken sich nicht auf den Anwender einer Klasse aus
+ Inkonsistente Zustände werden vermieden, da Zuständsänderungen nur über die Schnittstelle möglich sind
+ Einfachere Fehlersuche

- Schnittstelle muß sorgfältig definiert werden
- Kapselung hat mehr Operationsaufrufe zur Folge, was zu geringerer Performance führt

Zusammenfassung

Objekte

- **sind Abbildungen realer Dinge**
- **haben Verhalten, Zustand und Identität**
- **sind Instanzen genau einer Klasse**
- **kommunizieren über Nachrichten (Operationsaufrufe)**
  z.B. gibGeschw() ist die Botschaft an ein Autoobjekt, eine seiner Operationen auszuführen

Klassen

- **entstehen durch Zusammenfassung ähnlicher Objekte**
- **sind Abstraktionen, die nur das Wesentliche hervorheben**

Objekte und Klassen werden in UML dargestellt.

Kapselung hilft, lose gekoppelte Systeme zu entwerfen, bei denen Schnittstelle und Implementierung getrennt sind