11. Vererbung und Polymorphismus -

das fortgeschrittene Konzept

mit Klassen in Java

Siehe Kap. 9, Kap. 11, Kap. 10.4.4.

Kap. [illegible] 4.4 [illegible]

Zusätzlich Kap. 9.3.2 aus Buch.

Das Konzept der Vererbung

```
public class Adresse {
    public String name;
    public String strasse;
    public int hnr;
    public int plz;
    public String ort;
    public String mail;
    public String kom.
```

11.2

Walter Adresse erweitern.

public class FaxAdr. extends Adresse {

public String tel

public String fax ;

UML Diagramm in Abbildung 9.5.
Die Ist-ein Pfeil !

FaxAdr. übt die Eigenschaften von
Adresse. Vererbung. Umgekehrt ist
Adresse eine Generalisierung von FaxAdr.
FaxAdr. auch eine Spezialisierung
oder Erweiterung von Adresse.

Adresse Superklasse.
FaxAdr. Subklasse.

(1.39)

Polymorphismus - Überschreiben einer Methode der Superklasse in der Subklasse

Zusammenfassend die 4 Säulen der objektorientierten Programmierung in Abbildung 9.2.

Machen wir einige Beispiele:

Aufgabe 10.4

a) Keine Zugriffsmethode für Geburtsjahr da vorhanden. Kein direkter Zugriff auf die private Variable der Superklasse.

11.4

public Taxi (int z) {

super (z) // Konstruktor der
// Superklasse, statt
} // this auch möglich

public boolean validateNumber() {

Führe den Test mit der
Komponentenvariable nummer
durch und entsprechende Ausgabe }

Damit wird innerhalb (...)

public void setNummer (int n)

bei z.B. Nummer und

t = new Konstruktor Student( )

das neue validate Nummer genommen.

11.5

Die Instanz bestimmt dann, welche Methode genommen wird. Man erkennt den Sinn von Instanzmethoden.

Aufgabe 10.4

Die Klasse Klang:

public Klang(int b, int a)

ein Konstruktor. Beachte den Unterschied zu

public ref Klang;

```
public String toString()
    :
public void mehrPower(int b)
    :
```

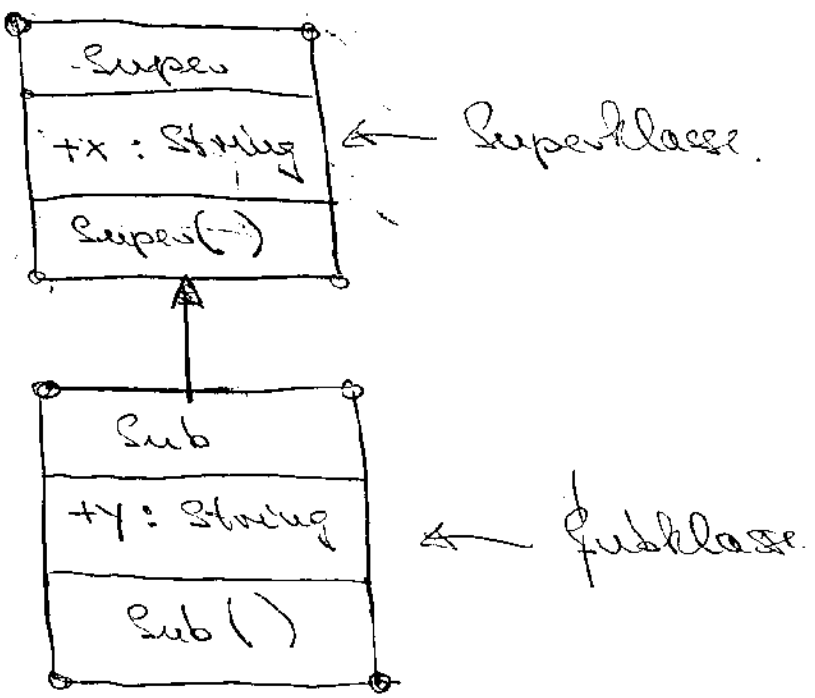
Das sind dann 2 Instanzmethoden.

Klasse Brad

```
Intelligenter Konstruktor
public Brade(int l, int b, int h) {
    super(b,h) // Entsprechender
               // Konstruktor der Superklasse.
}
```

Wie verhält es sich mit
der Objekterzeugung bei
Vererbung? Dazu noch einmal
Klasse 10.4.4. Ohne Vererbung
auf S. 10.24.

Situation jetzt:



Was passiert bei einem
Aufruf des Konstruktors der Subklasse

Sub Instanz = new Sub( )

① Speicherplatz für Instanzvariablen organisieren.

Hier heißt x und y.

② Defaultwerte für Instanzvariablen
nach Tabelle 10.2.

③ Aufgerufenen Konstruktor mit
Parametern ausführen.

Hier kann Sub( ) in der
Konstruktor von Sub ersten Zeile enthalten
→ this (...) oder
Von Super → super (...).

11.9

(a) 1. Zeile keine andere Konstruktor (nicht this(..) noch super(..)) so:
Wenn Superklasse vorhanden, Ergänzung des Konstruktors der Superklasse (Standard Konstruktor)

super();

wieder nach den Regeln (a), (b), (c).
Danach alle Initialisierungen der Klasse. Dann Rest des Konstruktors Sub().

Konstruktor nur einmal in erste Zeile.

(b) 1. Zeile super(...) so: Konstruktor der Superklasse, Dann Initialisierungen der Klasse, dann Rest des Konstruktors

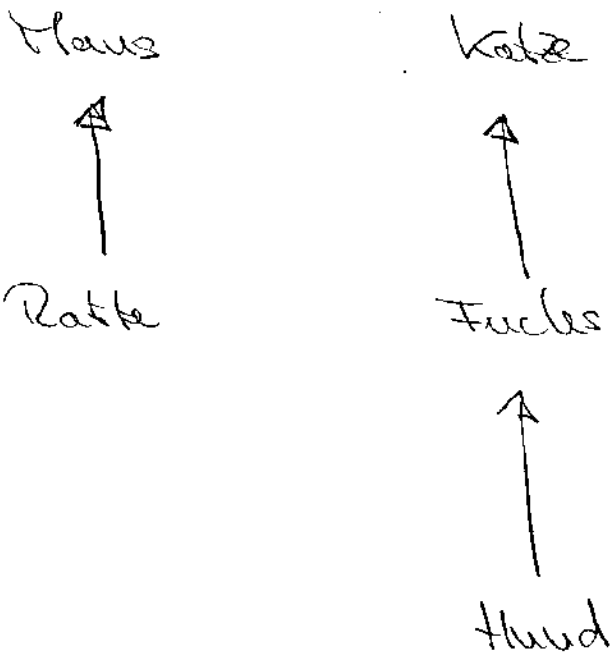
(c) 1. Zeile this(...) dann entsprechender Konstruktor, Dann Rest des Konstruktors.

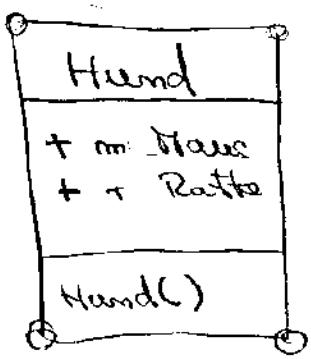
gemäß (c), der nächste Initialisierung.

Ausführung eines Konstruktors: Erst Initialisierungen der Deklaration, dann der Konstruktor selbst!

11.10

Aufgabe 10.5





UML Diagramm

(11.11)

Aufgabe 10.7

Fehler 1

Rückgabe null. (null ist Initialisierungswert für den Typ String).

Fehler 2

Alter als String wird gewandelt.

gibt die Adresse aus. Nicht die Werte der Instanzvariablen!

Fehler 3

gibt ein valides aus. oder Compilerfehler. wenn Fehler 2 nicht behoben.

Fehler 4 ✓

Fehler 5

Rückgabetyp void bei Konstruktor fehlt.

Fehler 6

Kein Defaultkonstruktor.

Was bewirkt nun new (Hund()) ? (11.11)

Hund() → Fuchs()

→ Katze

Ausgabe : Katze
Fuchs // wg. super
Maus // wg. Initialisierung
Maus // wg. Initialisierung
Ratte
Hund.

M.12

Fehler 4 ✓

Fehler 5

Rückgabetyp void bei Konstruktor falsch.

Fehler 6

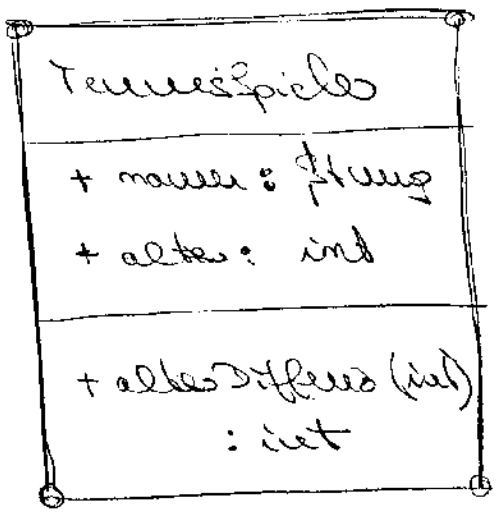
kein Defaultkonstruktor vorhanden.

public Fehler6() { }

ergänzen.

Aufgabe 10 &

a)



b) Instantiierung von main aus nicht möglich.

d) Über ein Objekt.

c) Wenn es so dann auch noch

Tennisspieler() { }.

g)

:

public TennisSpieler reifalge;

public int stabnummer;

:

h) public static int folgeNr.

k) Klassenvariablen dürfen nur auf Klassenvariablen zugreifen können der Main-Operator nicht. Es sei denn, es genommen selbst ein Objekt.

Weiter in Kapitel 11.

Nicht instanziierbar

public abstract class Wölfung

public abstract double dolBe()

// soll Wolf die Instanz

// in Dollar zurückgeben.

Abbildung 11.11 kurz für abstrakt.