Vererbung

- Definition von Klassen (*Unterklassen*), die von einer anderen Klasse (*Oberklasse*) abgeleitet sind
- Vererbung aller Datenelemente und Methoden von der Oberklasse an jede ihrer Unterklassen
- Erweiterung der Oberklasse durch neue Datenelemente und Methoden möglich
- (geerbte) Methoden mit derselben Signatur können in verschiedenen Unterklassen verschieden implementiert sein (*Polymorphie*)!

  $\longrightarrow$ realisiert durch *Überschreiben* von Methoden der Oberklasse
- keine Mehrfachvererbung (Erben von mehreren Oberklassen gleichzeitig), aber hierarchiches Erben möglich

Quellcodes einer Unterklasse (Prinzip)

```
public class Unterklasse extends Oberklasse {

    // neue Datenelemente

    // Konstruktoren
    public Unterklasse(...)  {
        super(...);  // optional
        // weitere Anweisungen;
    }

    // neue Methoden

    // überschriebene Methoden
    public void methode(...)  {
        super.methode(...);  // optional
        // weitere Anweisungen
    }

}
```

Konstanten

- werden mit dem Schlüsselwort `final` definiert
- der Wert konstanter Variablen kann nicht verändert werden
- von Klassen, die `final` sind, können keine Unterklassen abgeleitet werden
- Methoden, die `final` sind, können nicht überschrieben werden
- Klassenmethoden gelten implizit als `final`

Typumwandlung einfacher Datentypen

Typumwandlung	Syntax
Art: String → einfacher Datentyp Bsp.: `"17"` → `int`	int n; `n = Integer.parseInt("17");`
Art: einfacher Datentyp → String Bsp.: `2.83f` → `String`	String s; `s = String.valueOf(2.83f);`
Art: einf. Datentyp → einf. Datentyp Bsp: `2.83f` → `int`	*Cast-Konstrukt!* `int n = (int) 2.83f`

Ausnahme: `boolean`

Regeln beim Casting einfacher Datentypen

- Numerische Konvertierungen können mit Genauigkeitsverlust und/oder Vorzeichenwechsel einhergehen:
  - *Gleitkommatyp* → *Ganzzahltyp*: Abschneiden aller Kommastellen; ggf. resultiert der kleinste bzw. größte darstellbare Wert
  - *Ganzzahltyp* → *kleinerer Ganzzahltyp*: Abschneiden der höherwertigen Bits; Achtung: Vorzeichenwechsel möglich!
  - `double` → `float`: Wert mit kleinstmöglicher Differenz wird hergestellt
- Es werden keine Exceptions ausgelöst.
- Konvertierungen *ohne* explizites Casting sind möglich für:

  Ganzzahltyp → *größerer Ganzzahltyp*
  `float` → `double`
  `char` → `short`, `int` oder `long`

Schlüsselwörter zur Zugriffsmodifikation

Datenelemente/Methoden/Konstruktoren mit dem Modifier

`public`	sind für alle Klassen sichtbar;
`private`	sind nur für die Klasse sichtbar, in der sie vereinbart sind;
`protected`	sind für Klassen aus demselben Paket und für Unterklassen sichtbar.
ohne Modifier	sind für Klassen aus demselben Paket sichtbar

Diese Schlüsselwörter dürfen in der Definition von Methodenvariablen **nicht** auftreten!