

- 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*)!
—→ realisiert durch *Überschreiben* von Methoden der Oberklasse
- keine Mehrfachvererbung (Erben von mehreren Oberklassen gleichzeitig), aber hierarchisches Erben möglich

Quellcodes einer Unterklasse (Prinzip)

```
public class SubClass extends ParentClass {  
  
    // neue Datenelemente  
  
    // Konstruktoren  
    public SubClass(...) {  
        super(...); // optional  
        // weitere Anweisungen;  
    }  
  
    // neue Methoden  
  
    // überschriebene Methoden  
    public void method(...) {  
        super.method(...); // optional  
        // weitere Anweisungen  
    }  
}
```

- 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 \rightarrow einfacher Datentyp

Bsp.: "17" \rightarrow int

```
int n;
```

```
n = Integer.parseInt(17);
```

Art: einfacher Datentyp \rightarrow String

Bsp.: 2.83f \rightarrow String

```
String s;
```

```
s = String.valueOf(2.83f);
```

Art: einf. Datentyp \rightarrow einf. Datentyp

Bsp.: 2.83f \rightarrow int

Cast-Konstrukt!

```
int n = (int) 2.83f
```

Ausnahme: boolean

- 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

Regeln beim Casting einfacher Datentypen (2)

- Es werden keine Exceptions ausgelöst.
- Konvertierungen *ohne* explizites Casting sind möglich für:
 - Ganzzahltyp* → *größerer Ganzzahltyp*
 - float → double
 - char → short, int oder long

Datenelemente/Methoden/Konstruktoren mit dem Modifier

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

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