# II.2.1 Grundzüge der Objektorientierung

Freitag, 26. Oktober 2018 13:00

Arrays haben Vor- und Nachteile.

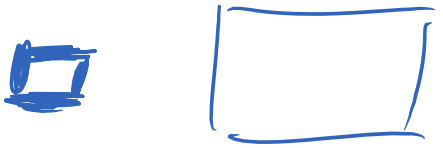
Vorteil: Wahlfreier Zugriff, d.h. man kann direkt auf jeden Array-Eintrag zugreifen.

Nachteile:

- Arrays fassen nur Elemente des gleichen Typs zusammen.
- Größe des Arrays liegt bei Erzeugung fest und kann nicht nachträglich geändert werden.
- Arrays können nur eine Eigenschaft eines modellierten Gegenstand ausdrücken.

Bsp: Rechteck hat Eigenschaften Laenge, Breite,

Strichstaerke ← int ↖ ↗ double



⇒ muss auf 3 Arrays verteilt

0. Rechteck:

laenge[0] = 2.5;
breite[0] = 2.0;
strichstaerke[0] = 1;

⇒ Zuweisung der Art

Rechteck0 = Rechteck1

braucht dann 3 Einzelzuweisungen

- Arrays sind reine Datenstrukturen. Prog.-Stücke zur Berechnung von Rechteck-Eigenschaften stehen außerhalb der Arrays.

⇒ Objektorientierung

# Klasse Rechteck

- neuer Datentyp Rechteck
- Jedes Objekt dieses Datentyps hat Eigenschaften:

  Attribute: laenge, breite, strichst.
  (laenge, breite: double; strichst.: int)

  Methoden: flaeche()
  (flaeche(): double)

- Man kann jetzt Variablen vom Datentyp Rechteck deklarieren.

```
int x;
Rechteck r;   <- danach ist r = null
:
```

- Erzeugung eines neuen Objekts mit "new".

```
r = new Rechteck();
```

- Zugriff auf Objekteigenschaften über r. ... :

```
r.laenge = 2.5;
r.breite = 2.0;
   :
```

r.flaeche() ergist 5.0

- "return" bedeutet: Beendigung der Methode und. Zurücklie- fern des Ergebnisses.
- "static": hängt nicht vom Objekt ab, gist es nur einmal in der Klasse.
  ⇒ laenge, ..., flaeche sind nicht static, denn r.laenge kann verschieden von s.laenge sein.

Jetzt werden alle Daten und Methoden eines Objekts zusam- men in einer Klasse realisiert.

Wenn r, s Rechtecke sind,

ist Zuweisung

r = s

möglich.

Variablen von Klassen-Datentypen sind wieder als Referenzvariablen realisiert (d.h. sie enthalten die Adresse auf dem Heap, an der das entspr. Objekt liegt).

⇒ Seiteneffekte
+ Garbage Collector
(wie bei Arrays)