

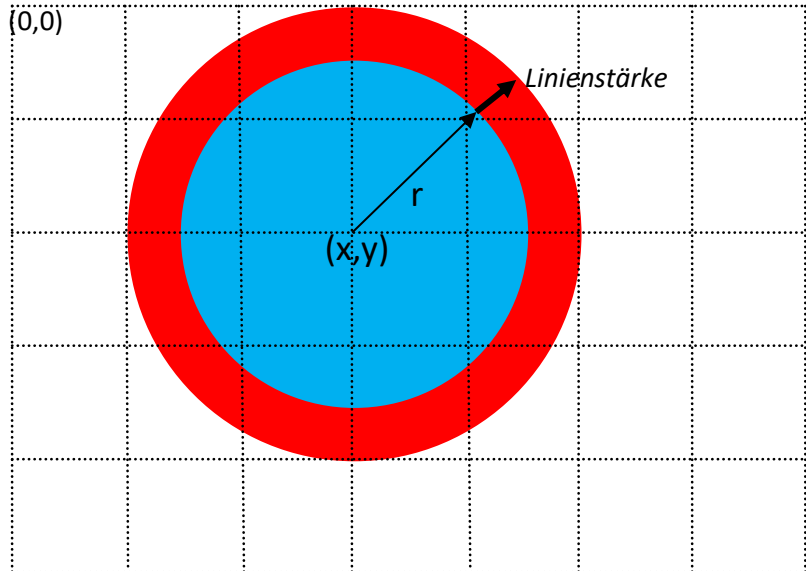
2.6.2 Objektorientierte Programmierung

Arbeitsblatt 03 Grundlegende Konzepte der Objektorientierten Programmierung

Assoziation

1. Implementiere ein Programm, das einen blau gefüllten Kreis mit dem Radius 100 Pixel und einem 30 Pixel breiten, roten Rand zeichnet.
 - Plane die Figur, indem du sie in dem Gitternetz rechts skizzierst.
 - Für die Position des Mittelpunkts werden die Attribute x und y , für den Durchmesser das Attribut d benötigt. Erstelle ein Klassen- und ein Objektdiagramm.

Klassendiagramm:



Objektdiagramm:

Objektdiagramm für das Zeichenfenster:

kreis1:Zeichenfenster

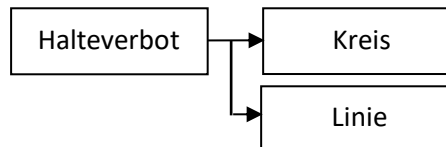
Breite=300
Höhe=300

- Codiere das Modell und speichere die Datei als `kreis1`.

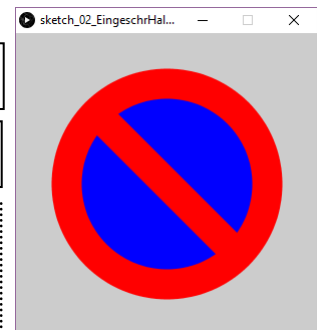
Hinweis: Die Anweisung zum Zeichnen eines Kreises lautet `ellipse(x, y, Breite, Höhe)`; wobei die Werte für Breite und Höhe gleich sein müssen. Für beide Werte kann also ein Attribut für den Durchmesser verwendet werden.

2. Das Verkehrszeichen „Eingeschränktes Halteverbot“ wird aus einem blau gefüllten Kreis mit rotem Rand sowie einer diagonal verlaufenden, roten Linie gezeichnet.

```
class EingHalteverbot {  
  EingHalteverbot() {  
  }  
  void zeichne() {  
    k1.zeichne();  
    l1.zeichne();  
  }  
}
```



Die Methode `zeichne()` der Klassen `Kreis` und `Linie` werden jetzt nicht in der Funktion `setup()` aufgerufen, sondern in der Methode `zeichne()` der Klasse `EingHalteverbot`. So wird eine Assoziation hergestellt.



- Erstelle das Programm in Processing (`EingeschrHalteverbot`).
3. Zusatzaufgabe: Ergänze eine zweite Linie von links unten nach rechts oben. Das daraus resultierende Verkehrszeichen bedeutet „Absolutes Halteverbot“ (`AbsolutesHalteverbot`).



2.6.2 Objektorientierte Programmierung

Arbeitsblatt 03 Grundlegende Konzepte der Objektorientierten Programmierung

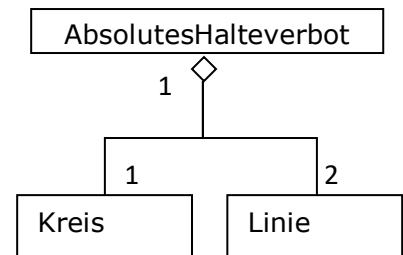
Aggregation

4. Um eine Aggregation herzustellen, werden die Objekte einer Klasse in dem Konstruktor einer anderen Klasse erstellt.
Für die Aggregation des Verkehrszeichens „Absolutes Halteverbot“ gilt: Die Klasse `AbsolutesHalteverbot` besteht aus einem `Kreis` und zwei `Linie`.

(Vorlagedatei: `v03_AbsolutesHalteverbot`)

- Ändere das Programm so ab, dass das Verkehrszeichen nicht als Assoziation, sondern als Aggregation gezeichnet wird.

```
class AbsolutesHalteverbot {
    Kreis k1;
    Linie l1;
    Linie l2;
    AbsolutesHalteverbot() {
        k1=new Kreis(150,150,200,30);
        l1=new Linie(80,80,220,220,30);
        l2=new Linie(80,220,220,80,30);
        k1.zeichne();
        l1.zeichne();
        l2.zeichne();
    }
}
```



*Zusammensetzung (Struktur)
des Verkehrszeichens in
einem Klassendiagramm*

Die Objekte `k1`, `l1` und `l2` der Klassen `Kreis` und `Linie` werden jetzt nicht in der Funktion `setup()` erzeugt, sondern in dem Konstruktor der Klasse `AbsolutesHalteverbot`. Damit sind die Objekte `k1`, `l1` und `l2` Teile davon, womit eine Aggregation vorliegt. Die Methode `zeichne()` wird hier nicht mehr benötigt.

Hinweise:

- Der Programmcode ist bereits vollständig vorhanden. Das lässt sich mit `ausschneiden()` und `einfügen()` bzw. den Tastenkombinationen `<Strg>+<X>` und `<Strg>+<V>` schnell erledigen.
- Die Klassen `Kreis` und `Linie` sind strukturierte Datentypen für die Attribute `k1`, `l1` und `l2` der Klasse `AbsolutesHalteverbot`:

AbsolutesHalteverbot
k1:Kreis l1:Linie l2:Linie

Kreis
x:int y:int d:int linie:int
zeichne ()

Linie
x1:int y1:int x2:int y2:int linie:int
zeichne ()

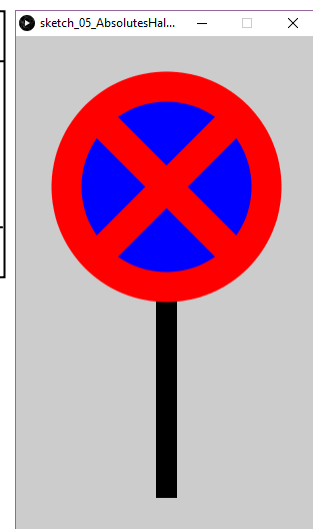
- Speichere das Programm unter `AbsolutesHalteverbotAggregation`.
5. Wenn man das Verkehrszeichen mit einem Pfosten zeichnet, könnte man von einem Verkehrsschild „Absolutes Halteverbot“ sprechen.
Der Pfosten kann als Rechteck gezeichnet werden.

- Ergänze die erforderlichen Klassen.
(`AbsolutesHalteverbotSchild`)

Hinweise:

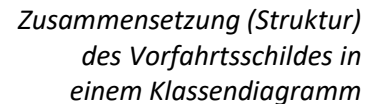
- Das Zeichenfenster muss nun etwas höher sein.
- Achte auf die Reihenfolge, in der die Objekte gezeichnet werden:
Der Pfosten soll von dem Verkehrszeichen verdeckt werden.

Pfosten
x:int y:int breite:int hoehe:int
zeichne ()




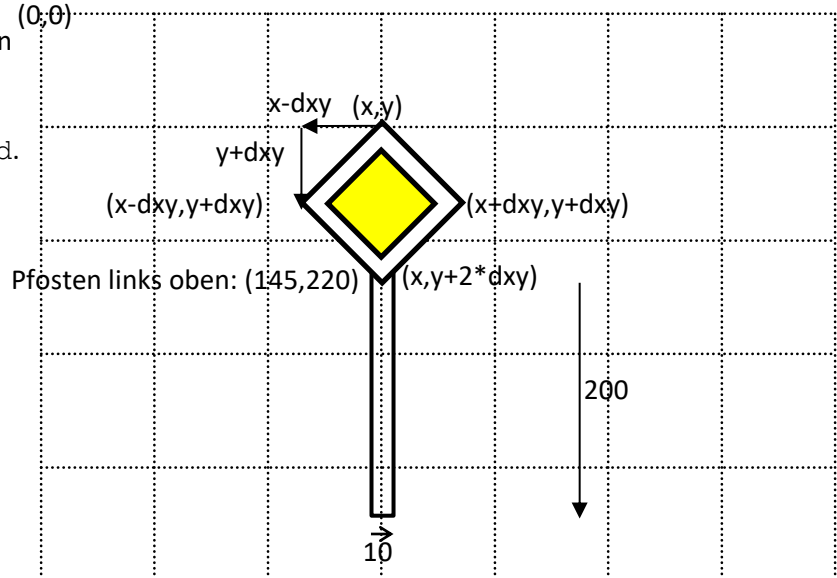


- Bei einer **Komposition** besteht eine Klasse *aus* weiteren Klassen bzw. *sind* Klassen Teil einer anderen Klasse, wobei die Teilklassen existenzabhängig von der Oberklasse sind. Im Klassendiagramm wird das dadurch gekennzeichnet, dass die Raute schwarz gefüllt wird. Hier werden nicht nur Objekte einer Klasse in dem Konstruktor einer anderen Klasse erstellt, sondern die Teilklassen komplett innerhalb der Oberklasse festgelegt.



- Hinweise:

- Die Aggregation wird hergestellt, indem du eine Klasse `Vorfahrtsschild` erstellst und alle weiteren Klassen innerhalb dieser Klasse einfügst. Letztlich sind also evtl. nur eine geschweifte Klammer zu versetzen und die jetzt in der Oberklasse enthaltenen Teilklassen einzurücken.
 - Die Objekte `p1`, `q1` und `q2` werden nicht in der Funktion `setup()` erstellt, sondern in dem Konstruktor der Klasse `Vorfahrtsschild`.
 - Für den Pfosten wird die neue Klasse `Pfosten` benötigt:
- 

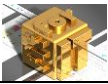


- Klassendiagramm für die Klasse `Vorfahrtsschild`:

- Ergänze das Objektdiagramm für p1:

p1:Pfosten

x=____
y=____
breite=____
hoehe=

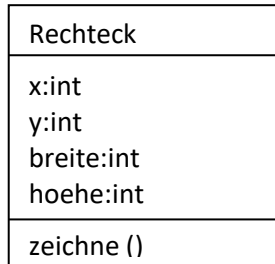


2.6.2 Objektorientierte Programmierung

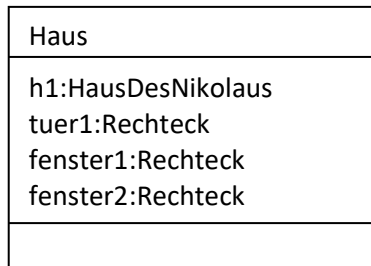
Arbeitsblatt 03 Grundlegende Konzepte der Objektorientierten Programmierung

7. In dem Programm `nikolaus1` sind eine Tür und zwei Fenster zu ergänzen (siehe Klassenstruktur rechts).
(Vorlagedatei: `v05_nikolaus1`)

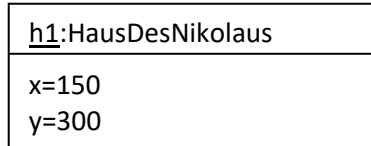
- Klassendiagramm für die Klasse `Rechteck`:



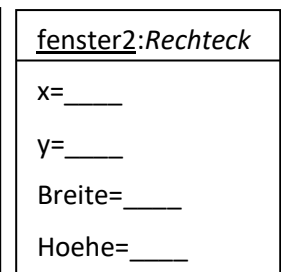
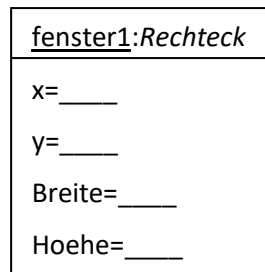
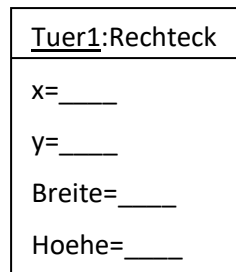
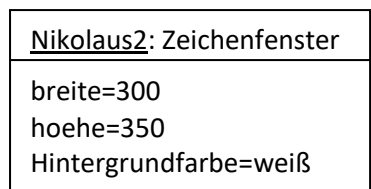
- Klassendiagramm für die Klasse `Haus`:



- Plane die Objekte, indem du in der Skizze rechts Koordinaten für die Eckpunkte der Rechtecke ergänzt. Für das Haus des Nikolaus gilt:



- Für die Füllfarbe grau gilt der Attributwert `110`.
- Verwende als Linienstärke 2 Pixel.
- Ergänze die Objektdiagramme zu den weiteren Objekten:



- Codiere die Ergänzungen in Processing und speichere das Programm als Version `nikolaus2`.

8. Zusatzaufgabe: Erstelle auf Basis des Programms `12_division` (vgl. Arbeitsblatt 01, S. 4, Nr. 12) eine Klasse `Rechner` mit den Kompositionen `Addierer`, `Multiplizierer` und `Dividierer`. Du kannst auch die Vorlagedatei `v06_division` verwenden (`14_komposition`). Erzeuge drei Objekte `r1`, `r2` und `r3` der Klasse `Rechner` mit unterschiedlichen Parametern.

