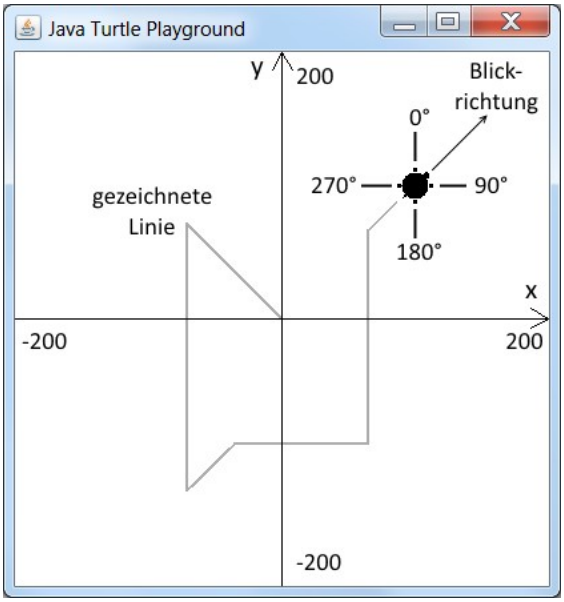
Die Java-Turtle



Mit der „Turtle“ kann man innerhalb eines Fensters zeichnen. Sie bewegt sich innerhalb eines Koordinatensystems, das seinen Ursprung in der Mitte des Fensters hat.

Die Turtle hat eine „Blickrichtung“ (engl. **heading**), die in Grad angegeben wird. heading = 45 bedeutet z.B., dass die Turtle 45° nach rechts oben schaut.

In diese Richtung bewegt sich die Turtle mit dem Befehl **forward**, und zeichnet dabei eine Linie. Dann kann sie mit **left** und **right** in eine andere Richtung gedreht werden, und zeichnet von dort mit dem Befehl forward eine weitere Linie, usw.

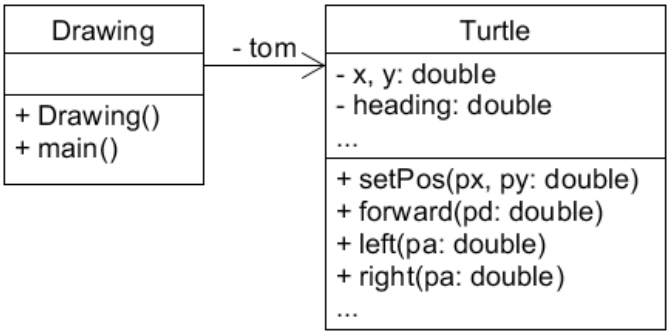
Programmbeispiel

```
import ch.aplu.turtle.*;

public class Drawing
{
   private Turtle tom;

   public Drawing()
   {
      tom = new Turtle();
   }

   public void main()
   {
      tom.forward(50.0);
      tom.left(90.0);
      ...
   }
}
```



Methoden der Klasse Turtle

Im Unterricht arbeiten wir mit der Turtle von Prof. Ägidius Plüss (aplu.ch), die eine umfangreiche Sammlung von Methoden bietet. Von diesen benötigen wir für die Aufgaben nicht alle, daher sind hier nur die wichtigsten aufgelistet. Wenn du Interesse hast, kannst du für weitere Anregungen in die Dokumentation schauen.

Konstruktoren

`tom = new Turtle();`	Erzeugt ein Objekt der Klasse Turtle in einem neuen Fenster mit Breite 400 und Höhe 400 Pixel.

`TurtleFrame frame;`
`Turtle tom;`
`...`
`frame = new TurtleFrame(`
`"Titel", 800, 600);`
`tom = new Turtle(frame);`
	Erzeugt ein Fenster („TurtleFrame“) mit Breite 800 und Höhe 600. Anschließend wird die Turtle „tom“ erzeugt und in das gerade erzeugte Fenster gesetzt.

Bewegung

`forward(double pDist)`	Bewegt Turtle in der aktuellen Richtung um pDist Pixel.
`back(double pDist)`	Bewegt Turtle rückwärts. Die Richtung ändert sich nicht.
`setPos(double px, double py)`	Setzt Turtle auf die Position (x, y), ohne zu zeichnen. Die Richtung der Turtle ändert sich nicht.
`left(double pAngle)` `right(double pAngle)`	Dreht Turtle von der aktuellen Richtung aus um pAngle nach links bzw. rechts (gemessen in Grad).
`heading(double pAngle)`	Setzt die Richtung der Turtle (egal wo sie vorher hinschaut).

Weitere Befehle

`hideTurtle()`	Macht Turtle unsichtbar (Turtle zeichnet schneller).
`showTurtle()`	Zeigt die Turtle wieder an.
`penUp()`	Hebt den Zeichenstift (Turtle bewegt sich, ohne zu zeichnen).
`penDown()`	Setzt Zeichenstift ab (Turtle zeichnet).
`penWidth(int pWidth)`	Setzt die Breite des Zeichenstifts (in Pixeln).
`setPenColor(String pColor) - Bsp: tom.setPenColor("blue");` Setzt die Stiftfarbe (d.h. die Farbe der Zeichnung) mithilfe eines String-Farbcodes. Es können z.B. red, blue, yellow, green, orange, purple, white, gray, black usw. benutzt werden. Vollständige Liste: `http://en.wikipedia.org/wiki/X11_color_names`	
`setPenColor(Color pColor) - Bsp: tom.setPenColor(new Color(0, 0, 255));` Setzt die Stiftfarbe. Um Color-Objekte zu erzeugen muss die Bibliothek java.awt.* importiert werden.	
`clear()`	Übermalt das Fenster mit weiß.
`clear(Color pColor)`	Übermalt das Fenster mit der Farbe pColor.

Zu den Aufgaben

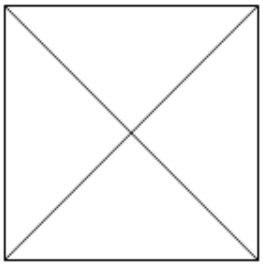
Nutze die bereitgestellte BlueJ-Vorlage.

Zu Beginn deiner Methoden rufe jeweils die Turtle-Methode clear() auf.
Setze die Turtle mit setPos() und heading() an ihre Anfangsposition. Programmiere die restliche Zeichnung dann **ohne** setPos() und heading(), nur mit forward(), left() und right().

Aufgabe 1

Die Methode aufgabe1() zeichnet den abgebildeten „Knick“. Die beiden Linien in der Mitte sollen so liegen, dass ihre Endpunkte ein gleichseitiges Dreieck ergeben. Die Länge der Linien wähle selbst.

Aufgabe 2



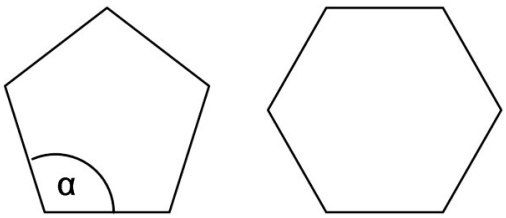
Die Methode aufgabe2() zeichnet ein Quadrat mit Diagonalen.

Die Linien sollen **nicht doppelt** gezeichnet werden. Um die Turtle zu bewegen, ohne dabei zu zeichnen, rufe vor der Bewegung die Methode penUp() auf, um den Stift „hochzuheben“, und setzte ihn anschließend mit penDown() wieder ab.

Die Länge der Diagonalen soll deine Methode berechnen (mithilfe des Satz des Pythagoras). Zur Berechnung der Wurzel nutze die Methode `double Math.sqrt(double pZahl)` – sqrt steht für „square root“.

Tipp: nutze eine Variable für die Seitenlänge und setze sie zu Beginn der Methode, z.B. auf 200.

Aufgabe 3



Die Methode aufgabe3(int pn, int pSeite) erhält zwei Parameter. Sie zeichnet ein regelmäßiges Vieleck mit pn Ecken und der Seitenlänge pSeite.

Tipp: Der Innenwinkel α eines regelmäßigen n-Ecks beträgt $\alpha = 180° - (360° : n)$
Bsp. für ein 5-Eck: $\alpha = 180° - (360° : 5) = 108°$