

Erinnerung: alle Objektvariablen und Methoden erhalten Sichtbarkeiten, wir kennen bisher „private“ und „public“.

21. Aufgabe (5 Punkte, equals, toString)

Kopieren Sie aus einem vorherigen Projekt die Klassen Linie und Punkt.

- Ergänzen Sie in der Klasse Linie eine passende Methode `public boolean equals(Linie other)` basierend auf der `equals`-Methode der Klasse Punkt, in der Sie Start- und Endpunkt auf Gleichheit prüfen. Zwei Linien, bei denen Start- und Endpunkt vertauscht sind, sollen nicht gleich sein. Es könnte hilfreich sein, erst b) zu lösen.
- Zeichnen Sie für Ihre `equals`-Methode der Klasse Linie ein Aktivitätsdiagramm.
- Ergänzen Sie in den Klassen Punkt und Linie Methoden `public String toString()`, so dass das folgende von der Veranstaltungsseite erhältliche Klasse nach Aufruf der Methode `testeLinieEquals` die rechts angegebenen Ausgaben produziert. Machen Sie ein Bildschirmfoto Ihrer Ausgabe.

```
public class Analyse{  
  
    public void testeLinieEquals(){  
        EinUndAusgabe io = new EinUndAusgabe();  
        Punkt p1 = new Punkt(1, 1);  
        Punkt p2 = new Punkt(9, 9);  
        Linie l1 = new Linie(p1, p2);  
        Linie l2 = l1;  
        Linie l3 = new Linie(p2, p1);  
        Linie l4 = new Linie(p1, null);  
        Linie l5 = new Linie(null, p2);  
        Linie l6 = new Linie(null, null);  
        Linie l7 = new Linie(new Punkt(1,1), new Punkt(9,9));  
        Linie l8 = new Linie(new Punkt(1,1), null);  
        Linie l9 = new Linie(null, new Punkt(9,9));  
        Linie la = new Linie(null, null);  
        io.ausgeben(l1 + "\n" + l2 + "\n" + l3 + "\n"  
            + l4 + "\n" + l5 + "\n" + l6 + "\n" + l7 + "\n");  
        io.ausgeben("l1 eq l1 : " + l1.equals(l1) + "\n");  
        io.ausgeben("l1 eq l2 : " + l1.equals(l2) + " - " + l2.equals(l1) + "\n");  
        io.ausgeben("l1 eq l3 : " + l1.equals(l3) + " - " + l3.equals(l1) + "\n");  
        io.ausgeben("l1 eq l4 : " + l1.equals(l4) + " - " + l4.equals(l1) + "\n");  
        io.ausgeben("l1 eq l5 : " + l1.equals(l5) + " - " + l5.equals(l1) + "\n");  
        io.ausgeben("l1 eq l6 : " + l1.equals(l6) + " - " + l6.equals(l1) + "\n");  
        io.ausgeben("l1 eq l7 : " + l1.equals(l7) + " - " + l7.equals(l1) + "\n");  
        io.ausgeben("l4 eq l8 : " + l4.equals(l8) + " - " + l8.equals(l4) + "\n");  
        io.ausgeben("l5 eq l9 : " + l5.equals(l9) + " - " + l9.equals(l5) + "\n");  
        io.ausgeben("l6 eq la : " + l6.equals(la) + " - " + la.equals(l6) + "\n");  
        io.ausgeben("la eq null : " + la.equals(null) + "\n");  
    }  
}
```

```
Blue! Konsole - AufgabeLiniePunktMitEqToStr  
Optionen  
von (1,1) nach (9,9)  
von (1,1) nach (9,9)  
von (9,9) nach (1,1)  
von (1,1) nach null  
von null nach (9,9)  
von null nach null  
von (1,1) nach (9,9)  
l1 eq l1 : true  
l1 eq l2 : true - true  
l1 eq l3 : false - false  
l1 eq l4 : false - false  
l1 eq l5 : false - false  
l1 eq l6 : false - false  
l1 eq l7 : true - true  
l4 eq l8 : true - true  
l5 eq l9 : true - true  
l6 eq la : true - true  
la eq null : false
```

22. Aufgabe (4 Punkte, einfache Methoden mit if)

Zur Veranschaulichung der Ergebnisse der beiden Teilaufgaben können von der Veranstaltungsseite die Programme `tagExistiert.exe` sowie `dreieckanalyse.exe` jeweils in einem Zip-File geladen und in Windows 10 ausgeführt werden.

Schreiben Sie eine Klasse `Ueberpruefung`, die nur eine Objektvariable vom Typ der Klasse `EinUndAusgabe` zum Einlesen und Ausgeben nutzt. Schreiben Sie die folgenden Methoden.

- a) Schreiben Sie eine Methode `tagExistiert()` mit folgendem Ablauf. Eine Person gibt beliebig den Tag, den Monat und das Jahr ein und die Methode gibt als Text auf dem Bildschirm aus, ob der Tag existiert. Man erkennt Schaltjahre daran, dass sie durch vier teilbar sind. Falls das Jahr auch durch 100 teilbar ist, muss es allerdings auch durch 400 teilbar sein, um ein Schaltjahr zu sein (2000 und 2400 sind Schaltjahre, 2100, 2200 und 2300 nicht). Die rechte Seite zeigt zwei Aufrufe, Eingaben sind blau. Natürlich sollen auch Eingaben von Werten wie (3,13,2022) zu Ausgabe „Tag existiert nicht“ führen.

```
BlueJ: Konsole - AufgabeUeberprueft
Optionen
Tag: 29
Monat: 2
Jahr: 2020
Tag existiert
Tag: 29
Monat: 2
Jahr: 2100
Tag existiert nicht
```

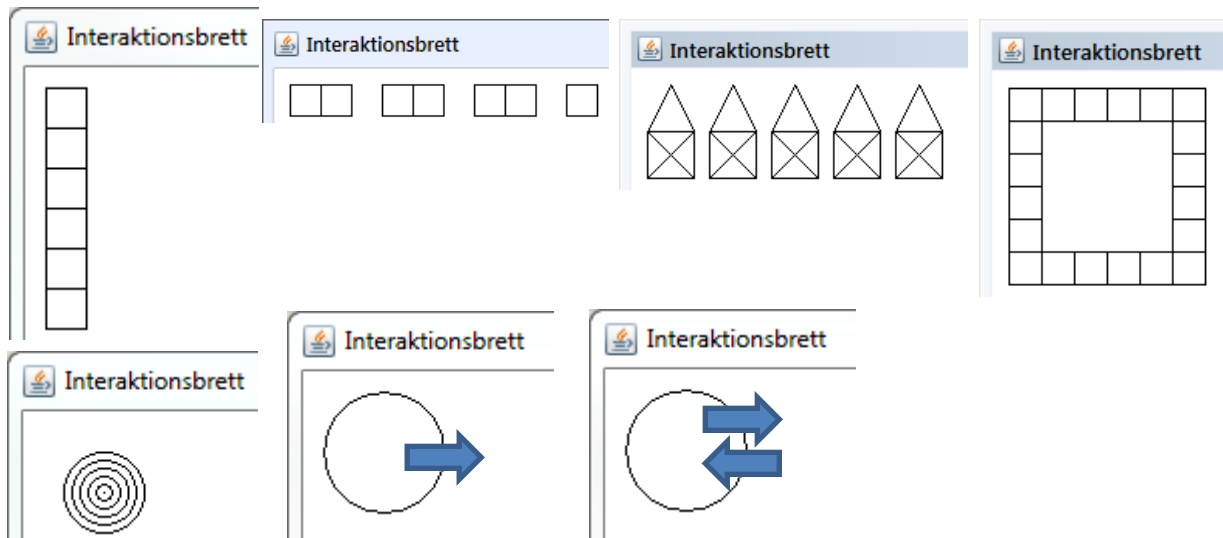
- b) Eine Methode `public void dreieckanalyse()` soll drei ganzzahlige Werte durch eine Person eingeben lassen, die für die Seitenlängen eines Dreiecks stehen sollen. Für diese drei Werte a, b, und c werden dann folgende Überprüfungen durchgeführt:
Wenn einer der drei Werte nicht größer als 0 ist oder aus den Seiten kein Dreieck konstruiert werden kann, erhält der Nutzer einen Hinweis.
Falls aus den Werten ein Dreieck konstruiert werden kann, werden die Eigenschaften „ist gleichschenkelig“ (wenn zwei Seiten gleich lang), „ist gleichseitig“ (wenn alle Seiten gleich lang), „ist rechtwinklig“ (wenn einer der drei Winkel des Dreiecks 90° beträgt, es muss „ $a^2+b^2 = c^2$ “ gelten, wobei c immer die längste Seite sein muss, die Sie erst finden müssen) ausgegeben. Realisieren Sie das Programm so, dass alle Eigenschaften mit in einzelnen Booleschen Methoden geprüft werden, z. B. `private boolean istGleichseitig(int,int,int)`, die Sie dann nutzen.

```
BlueJ: Konsole - AufgabeUeberpruefungDreieckTagLoesung
Optionen
Laenge der ersten Seite: 9
Laenge der zweiten Seite: 4
Laenge der dritten Seite: -1
Alle Seitenlaengen muessen positiv sein!
Laenge der ersten Seite: 5
Laenge der zweiten Seite: 5
Laenge der dritten Seite: 10
Dreieck nicht konstruierbar!
Laenge der ersten Seite: 5
Laenge der zweiten Seite: 5
Laenge der dritten Seite: 5
ist gleichschenkelig
ist gleichseitig
Laenge der ersten Seite: 3
Laenge der zweiten Seite: 4
Laenge der dritten Seite: 5
ist rechtwinklig
Laenge der ersten Seite: 6
Laenge der zweiten Seite: 4
Laenge der dritten Seite: 5
normales Dreieck
```

23. Aufgabe (6 Punkte, while-Schleifen)

Schreiben Sie eine Klasse `Dekoration`, die eine Objektvariable vom Typ `Interaktionsbrett` nutzt und die folgenden Methoden implementiert.

- Mit Hilfe einer `while`-Schleife werden in einer Methode sechs Quadrate untereinander gezeichnet.
- Mit Hilfe einer `while`-Schleife werden 10 Quadrate nebeneinander gezeichnet, wobei jedes dritte Quadrat fehlt (damit sind es nur sieben Quadrate).
- Mit Hilfe einer `while`-Schleife werden in einer Methode fünf Nikolaushäuser nebeneinander gezeichnet.
- Mit Hilfe *einer* `while`-Schleife wird in einer Methode das Muster mit dem umrahmten Quadrat gezeichnet.
- Mit Hilfe einer `while`-Schleife werden um einen Punkt fünf Kreise gezeichnet.



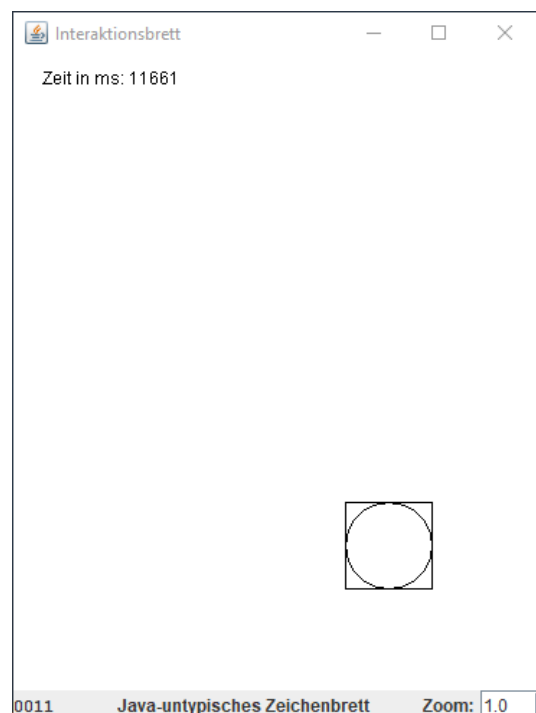
- f) Mit Hilfe einer while-Schleife bewegt sich ein Kreis von links nach rechts über den Bildschirm. Bauen Sie zwischenzeitlich kleine Pausen ein, das Interaktionsbrett hat eine Methode dazu. (Den blauen Pfeil sollen Sie natürlich nicht zeichnen, er zeigt nur die Bewegungsrichtung.)
- g) Erweitern Sie f) in einer neuen Methode so, dass der Kreis sich am Ende wieder von rechts nach links zur Ausgangsposition bewegt. Dies soll sich solange wiederholen, bis der Nutzer das Interaktionsbrett schließt. Nutzen Sie dazu eine Endlosschleife `while(true)` und klären Sie in der Schleife, z. B. mit Booleschen Variablen, in welche Richtung sich der Kreis bewegt. *Es darf nur die genannte eine while-Schleife in dieser Methode auftreten.*

24. Aufgabe (3 Punkte, reaktive Programme)

Zur Veranschaulichung des Ergebnisses der letzten Teilaufgabe kann von der Veranstaltungsseite das Programm `reaktiveQuadrate.exe` in einem Zip-File geladen und in Windows 10 ausgeführt werden. Da das Programm nicht in Java geschrieben ist, sieht das Interaktionsbrett etwas anders aus. Das Programm wird mit einem Klick auf „X“ rechts-oben beendet.

Reaktive Programme sollen unmittelbar auf Nutzungsaktionen, wie Tastatureingaben und Mausbewegungen reagieren. Dies ist auch mit dem Interaktionsbrett möglich. Lesen Sie sich dazu den angehängten Text zur reaktiven Programmierung durch und laden Sie sich das Projekt zur Aufgabe, das bereits ein Beispielprogramm enthält, herunter. Mit dem Beispielprogramm kann ein Quadrat mit den Tasten „w“, „a“, „d“, „x“ über den Bildschirm gesteuert werden.

- a) Im Beispielprogramm befindet sich eine Methode `richtung()`. Beschreiben Sie mit einem Text wozu diese Methode da ist.
- b) Ergänzen Sie die Klasse so, dass man mit den Tasten „q“, „e“, „y“, „c“ auch schräg laufen kann.



- c) Das Interaktionsbrett kann mit der Methode `zufall(min,max)` eine zufällige Zahl zwischen den int-Werten `min` und `max`, jeweils einschließlich, zurückgeben. Nutzen Sie dies, um einen Kreis zufällig auf dem Bildschirm zu zeichnen. Messen Sie dann die Zeit, die benötigt wird, bis das Quadrat auf den Kreis geschoben wurde und geben Sie diese Zeit aus, wie in der Abbildung zur Aufgabe gezeigt. Es kann sinnvoll sein das Quadrat in einem Schritt gleich mehrere Pixel zu bewegen.
- d) Erweitern Sie das Programm so, dass fünfmal das Rechteck auf einen Kreis geschoben werden muss. Hat das Rechteck dabei einen Kreis erreicht, wird ein neuer Kreis an einer anderen Position gezeichnet.

Hintergrund: Reaktive Programmierung mit der Klasse `Interaktionsbrett`

Eine Software ist ein reaktives System, wenn es seine Berechnungen nach dem Start nicht selbstständig ausführt, sondern immer wieder von außen angestoßen wird. Grafische Oberflächen sind ein Beispiel für ein reaktives System, da z. B. erst etwas ausgeführt wird, wenn z. B. bei der Nutzung ein Knopf angeklickt wird.

Bei der Programmierung ist die Reaktion auf eine solche Aktion zu programmieren. Typischerweise entstehen dazu Methoden, die vom umgebenden System (z. B. Betriebssystem oder virtuelle Maschine) aufgerufen werden. Damit das System weiß, welche Methoden genutzt werden, muss dies „irgendwo“ vereinbart sein.

Das Interaktionsbrett hat u. a. die Möglichkeit, auf Tastatureingaben zu reagieren und diese Information an interessierte Objekte weiterzugeben. Dazu wird über die folgende Methode ein Objekt beim Interaktionsbrett angemeldet, das über Tastatureingaben informiert werden soll. Dabei können bei einem Parameter vom Typ `Object` Objekte beliebiger Klassen übergeben werden, also auch beliebige selbst geschriebene.

Method Summary

<code>void</code>	<code>willTasteninfo</code> (<code>java.lang.Object o</code>) Objekte können an ein Interaktionsbrett so übergeben werden, dass sie informiert werden, wenn eine Taste gedrückt wurde.
-------------------	---

Das Objekt, das informiert werden soll, muss die folgende Methode realisieren und erhält die gedrückte Taste als `String` zurück.

```
public void tasteGedruickt(String s)
```

Wird z. B. das Objekt selbst mit `this` übergeben, das die Methode `willTasteninfo()` ausführt, (`ib.willTasteninfo(this)`), muss in der Klasse des Objekts auch diese Methode realisiert werden. Die detaillierte Dokumentation auf der Web-Seite zeigt mögliche Werte, die das Objekt über den `String`-Parameter erhalten kann, u. a. auch Funktionstasten (<http://kleuker.iui.hs-osnabrueck.de/querschnittlich/Interaktionsbrett/Version1/index.html>). (Auf Apple-Rechnern werden eventuell nicht alle Sondertasten unterstützt, generell findet das Praktikum unter Windows 10 statt.)

Weitere Informationen zum Interaktionsbrett sind auch <https://youtu.be/vGryrMcNj6s> (38:04) zu entnehmen.