Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thüring
Dipl.-Inform. Steffen Schiele
Dipl.-Inform. Andreas Bienert

Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik

Wintersemester 2014/15

Objektorientierte Programmierung: Hausaufgabenblatt 6

Abgabe: 04.12.2014, 12:00 — Übung: 08./09.12.2014

Hausaufgabe 6: (4 Theoriepunkte)

"Spirograph" ist ein Zeichengerät, mit dem man "Hypozykloiden" zeichnen kann. Man hat dazu eine Schablone mit einem kreisförmigen Loch, in dem ein Innen-Zahnkranz ist. Dieser Kreis habe den Radius R. Außerdem hat man eine kleinere kreisförmige Schablone mit einem Außen-Zahnkranz. Diese habe den Radius r. In der kleineren Schablone sind mehrere Löcher. Man wählt eins mit dem Abstand a vom Mittelpunkt des kleineren Kreises, steck dort einen Stift hinein, und rollt den kleineren Kreis im größeren Kreis ab. Dadurch entstehen die besagten Kurven, und zwar unterschiedliche, je nach dem Verhältnis der drei Parameter. Die Formeln für die x- und y-Koordinaten in Abhängigkeit von der Zeit t sind:

$$x(t) = (R-r)\cos(\frac{r}{R}\,t) + a\cos\Big((1-\frac{r}{R})\,t\Big)$$

$$y(t) = (R-r)\sin(\frac{r}{R}\,t) - a\sin\Big((1-\frac{r}{R})\,t\Big)$$

Da sin und cos periodisch mit Periondenlänge 2π sind, reicht es jedenfalls, den Parameter t bis $R{*}2\pi$ laufen zu lassen. Tatsächlich reicht sogar die unter a) berechnete Schlaufenanzahl als Faktor vor 2π. Es ist auch klar, dass die Ergebniswerte betragsmäßig jedenfalls nicht größer als $(R-r)+a$ werden können (das ist später für die Skalierung wichtig). Mehr Informationen zur Herleitung der Formel und einige Beispiel-Kurven finden Sie unter:

`http://www.mathematische-basteleien.de/spirograph.htm`

a) Schreiben Sie eine statische Methode `zyklen` mit den Parametern

- `R` (ganze Zahl, Radius des äußeren Kreises), und
- `r` (ganze Zahl, Radius des inneren Kreises).

Diese Methode soll die Anzahl der "Schlaufen" der Kurve liefern (eine ganze Zahl). Die Formel dafür ist:

$$n = \frac{R}{\mathrm{ggT}(R,r)}$$

Dabei ist $\mathrm{ggT}(R,r)$ der größte gemeinsame Teiler von R und r. Zur Berechnung des ggT verwendet man üblicherweise den Euklischen Algorithmus. Die klassische Variante zieht solange die jeweils kleinere Zahl von der größeren ab, bis beide Zahlen gleich

sind. Dann ist das Ergebnis der ggT. Die Korrektheit beruht auf der Tatsache, dass für $a > b$ gilt $\mathrm{ggT}(a,b) = \mathrm{ggT}(b, a-b)$ (der ggT ist ja als Faktor in beiden Zahlen und auch in der Differenz enthalten). Beispiel: ggT von 12 und 20: Man berechnet die Differenz $20 - 12 = 8$. Anschließend $12 - 8 = 4$. Als nächstens $8 - 4 = 4$. Nun sind beide Zahlen gleich und 4 ist das Ergebnis.

b) Definieren Sie eine statische Methode `hypozykloide` (ohne Rückgabewert) mit folgenden Parametern:

- `R`, eine ganze Zahl, der Radius des äußeren Kreises,
- `r`, eine ganze Zahl, der Radius des inneren Kreises,
- `a`, eine ganze Zahl, der Abstand des Stiftes vom Mittelpunkt des inneren Kreises,
- `xArr`, ein Array der Größe 1000 von `int`-Werten,
- `yArr`, ein Array der Größe 1000 von `int`-Werten.

Ihre Methode soll in die beiden übergebenen Arrays 1000 Koordinatenpaare der Kurve schreiben, wobei die Koordinaten zwischen `-200` und `+200` (einschließlich) liegen müssen. Das Hauptprogramm zeichnet dann einen Streckenzug durch diese Punkte.

Hinweise:

- Sei n die unter a) berechnete Anzahl Zyklen, also das Ergebnis des Aufrufes `zyklen(R, r)`. Um 1000 Werte für t zu generieren, verwenden Sie die Formel $t_i = i*(n*2\pi/999)$ mit $i = 0,\ldots,999$. Selbstverständlich hat `t` den Typ `double`.
- Damit die Koordinaten in dem gewünschten Bereich liegen, multiplizieren Sie das Ergebnis der obigen Formeln mit $200/((R-r)+a)$.
- Einen `double`-Wert `x` können Sie mit `(int) Math.round(x)` in ein `int` umwandeln. Der Typ-Cast nach `int` ist nötig, weil `Math.round` ein Ergebnis vom Typ `long` liefert (einem größeren ganzzahligen Typ als `int`).
- Die Konstante π ist in Java als `Math.PI` definiert. Entsprechend die Sinus-Funktion als `Math.sin` und Kosinus als `Math.cos`.

Fügen Sie Ihre Methoden in folgendes Rahmenprogramm ein:

`http://www.informatik.uni-halle.de/~brass/oop14/homework/Kurve.java`

Es ist auch unten abgedruckt. In der `main`-Methode finden Sie den Aufruf der Methoden, die Sie entwickeln sollen (falls es wegen der Parameter irgendwelche Unklarheiten gibt). Sie dürfen aber das Rahmenprogramm nicht ändern, sondern nur Ihre Methoden hinzufügen (es sei denn, Sie würden einen Fehler finden). Dieses Programm bekommt seine Eingaben (die Parameter R, r und a) aus der Kommandozeile. Ein möglicher Aufruf ist also:

```
java Kurve 20 14 8
```

Sie können es auch ohne Kommandozeilen-Argumente aufrufen, dann werden Standardwerte (`5 1 4`) benutzt.

```java
// OOP - Hausaufgabe 6: Hypozykloide
// Name:

import java.awt.*;
import java.awt.event.*;

class Kurve extends Frame implements WindowListener {

	// Fenster-Groesse:
	private static final int X_SIZE = 500;
	private static final int Y_SIZE = 500;

	// Maximale Kooridinaten (positiv und negativ):
	private static final int XY_MAX = 200;

	// Anzahl Punkte:
	private static final int POINTS = 1000;

	// Konstruktor (erzeugt Fenster-Objekt):
	public Kurve(int[] xArr, int[] yArr, int cycles) {
		super("Hypozykloide␣(" + cycles + "␣Zyklen)");
		xCoords = xArr;
		yCoords = yArr;
		setBackground(Color.WHITE);
		setSize(X_SIZE, Y_SIZE);
		setVisible(true);
		addWindowListener(this);
	}

	// Verschiedene Fenster-Ereignisse (ohne Reaktion):
	public void windowActivated(WindowEvent e) {}
	public void windowDeactivated(WindowEvent e) {}
	public void windowIconified(WindowEvent e) {}
	public void windowDeiconified(WindowEvent e) {}
	public void windowOpened(WindowEvent e) {}

	// Benutzer will Fenster schliessen:
	public void windowClosing(WindowEvent e) {
		e.getWindow().setVisible(false);
		e.getWindow().dispose();
	}
	// Wenn Fenster geloescht ist, endet das Programm:
	public void windowClosed(WindowEvent e) {
		System.exit(0);
	}

	// Attribute fuer zu zeichnende Koordinaten:
	private int xCoords[];
```



```java
	private int yCoords[];

	// Fehlermeldung und Schluss:
	private static void error(String msg) {
		System.out.println("Fehler:␣" + msg);
		System.exit(1);
	}

	// Hier wird gezeichnet:
	@Override public void paint(Graphics g) {

		// Fensterrahmen beruecksichtigen:
		int left = getInsets().left;
		int right = getInsets().right;
		int xCenter = (X_SIZE - left - right) / 2 + left;
		int top = getInsets().top;
		int bottom = getInsets().bottom;
		int yCenter = (Y_SIZE - top - bottom) / 2 + top;

		// Farbe des Zeichenstifts:
		g.setColor(Color.red);

		// Linienzug zeichnen:
		for(int i = 1; i < xCoords.length; i++) {
			int x_von = xCoords[i-1] + xCenter;
			int y_von = yCoords[i-1] + yCenter;
			int x_bis = xCoords[i]   + xCenter;
			int y_bis = yCoords[i]   + yCenter;
			g.drawLine(x_von, y_von, x_bis, y_bis);
		}
	}

	// Hauptprogramm:
	public static void main(String[] args) {
		// Default-Werte fuer Parameter:
		int R = 5;
		int r = 1;
		int a = 4;

		// Argumente aus der Kommandozeile:
		if(args.length > 3)
			error("Zu␣viele␣Argumente!");
		try {
			if(args.length >= 1)
				R = Integer.parseInt(args[0]);
			if(args.length >= 2)
				r = Integer.parseInt(args[1]);
			if(args.length >= 3)
```

```java
                a = Integer.parseInt(args[2]);
        }
        catch(Exception e) {
            error("Argument␣keine␣ganze␣Zahl!");
        }
        if(R <= 0 || r <= 0)
            error("R␣und␣r␣muessen␣positiv␣sein!");
        if(a < 0)
            error("a␣darf␣nicht␣negativ␣sein!");
        if(r > R)
            error("Es␣muss␣r␣<=␣R␣gelten!");
        if(R == r && a == 0)
            error("Argumente␣liefern␣nur␣Punkt!");

        // Schlaufen/Zyklen berechnen:
        int n = zyklen(R, r);

        // Koordinaten berechnen und pruefen:
        int[] xArr = new int[POINTS];
        int[] yArr = new int[POINTS];
        hypozykloide(R, r, a, xArr, yArr);
        for(int i = 0; i < POINTS; i++) {
            if(xArr[i] < -XY_MAX || xArr[i] > XY_MAX)
                error("xArr["+i+"]␣=␣"+xArr[i]);
            if(yArr[i] < -XY_MAX || yArr[i] > XY_MAX)
                error("yArr["+i+"]␣=␣"+yArr[i]);
        }

        // Ergebnisse anzeigen:
        new Kurve(xArr, yArr, n);
    }

    // Hier Ihre Methode zur Berechnung der Anzahl Zyklen:

    // Hier Ihre Prozedur zur Berechnung der Koordinaten:
}
```


Übungsaufgabe 6A: (ohne Abgabe)

Bitte bearbeiten Sie die Übungsaufgaben auf dem Hausaufgabenblatt, aber geben Sie diese Aufgaben nicht ab. Diese Aufgaben werden in der Übung besprochen. Sie müssen Ihre Lösung eventuell in der Übung vorführen.

Vereinfachen Sie folgende Programm-Ausschnitte:

a)

```java
if(a < b) {
    if(c == true)
        n = 1;
}
```

b)

```java
if(a < b && d == 1) {
    n = 1;
}
if(a < b && d == 2) {
    n = 1;
}
```

c)

```java
while(a < b) {
    if(a == b)
        System.out.println("gleich!");
    a++;
}
if(a < b)
    System.out.println("kleiner!");
```

d)

```java
while(a < b) {
    System.out.println(a);
    if(a < b)
        break;
    a++;
}
```

e)

```java
if(a < b) {
    n = 1;
    m = 2;
}
else {
    m = 2;
    n = 1;
}
```

Übungsaufgabe 6B: (ohne Abgabe)

Was gibt das folgende Programm aus?

```
class H6B {
        static void f(int n) {
                n = 5;
        }

        static void g(int[] a) {
                int n = 4;
                a[0] = n;
        }

        public static void main(String[] args) {
                int n = 3;
                f(n);
                System.out.println(n);
                int[] a = new int[3];
                System.out.println(a[0]);
                a[0] = 7;
                f(a[0]);
                System.out.println(a[0]);
                g(a);
                System.out.println(a[0]);
                a[0] = n;
                g(a);
                System.out.println(n);
        }
}
```

Auch diese Aufgabe bitte nicht einsenden, aber bis zur Übung bearbeiten.