Hausaufgabe 9: Vererbung Abgabe: 13.12.2016

Aufgabe 9.1: Geometrische Figuren

In dieser Aufgabe sollen geometrische Figuren grafisch auf dem Bildschirm angezeigt werden. Laden Sie sich dazu bitte das Java-Programm `Graphic.java` von der Internet-Seite der Veranstaltung runter. Dieses Programm ist ein Rahmen, um ihre geometrischen Figuren auf dem Bildschirm anzeigen zu können. Es besteht aus den Klassen `Graphic` und `Figure`.
`Graphic` hat die folgenden Methoden und Konstruktoren:

```
public Graphic(int width, int height)
/* Oeffnet ein Grafikfenster mit einer Zeichenflaeche der
   angegebenen Hoehe und Breite */

public void add(Figure f)
/* Fuegt die angegebene geometrische Figur in die
   Zeichenflaeche ein */

public void remove(Figure f)
/* Entfernt die geometrische Figur aus der Zeichenflaeche */

public void update(int ms)
/* Aktualisiert die Zeichenflaeche. Aenderungen werden erst
   sichtbar, nachdem diese Methode aufgerufen wurde. Wartet
   vorher ms Millisekunden. Damit ist eine einfache
   Animation moeglich */
```

Ihre geometrischen Klassen müssen von der Klasse `Figure` ableiten. `Figure` besitzt eine `ArrayList` von Punkten (die Klasse `Point` ist eine Java-Klasse aus dem Paket `java.awt`). Sie kann in abgeleiteten Klassen direkt verändert werden. In der Grafik werden Verbindungslinien zwischen den jeweils benachbarten Punkten der Liste gezeichnet.

Außerdem besitzt `Figure` die Methode

```
shift(int x, int y)
```

die alle Punkte aus der ArrayList um den angegebenen Wert verschiebt.

Testen Sie die Klassen anhand des folgenden Beispiel-Codes aus und machen Sie sich die Funktionsweise klar:

```
import java.awt.*;
//Linie zwischen zwei Punkte
public class Line extends Figure {
  public Line(Point x1, Point x2) {
    points.add(x1);
    points.add(x2);
  }
}
```

```
public class Test {
  public static void main(String[] args) {
    Graphic g = new Graphic(300,300); //Groesse der Grafik

    //Linie von (10/10) nach (200/100)
    //Der Koordinatenursprung ist links oben
    Line l = new Line(new Point(10,10), new Point(200,100));
    g.add(l);
    g.update(0);
  }
}
```

a) Schreiben Sie eine Klasse für ein Rechteck. Fügen Sie folgende Konstruktoren und Methoden hinzu:

```
public Rectangle(Point p1, Point p2)
/* Konstruktor mit zwei gegenueberliegenden Eckpunkten */

public Rectangle(Rectangle r) /* Copy-Konstruktor */

public void scale(double d)
/* Skaliert das Rechteck um den Faktor d.
   Der Mittelpunkt des Rechtecks bleibt erhalten */

public Rectangle uniteWith(Rectangle r)
/* Gibt das kleinstmoegliche Rechteck zurueck,
   dass sowohl this als auch r enthaelt */
```

b) Schreiben Sie eine Klasse `Circle` für einen Kreis. Konstruieren Sie einen Kreis aus 100 kleinen Sekanten. Benutzen Sie die Sinus- und Cosinus-Funktion zur Berechnung der Endpunkte der Sekanten.

Die Klasse soll zwei Konstruktoren besitzen. Der Erste soll nur den Mittelpunkt als Parameter erhalten und den Radius per Default auf 100 setzen. Der Zweite soll sowohl den Mittelpunkt, als auch den Radius per Übergabeparameter erhalten.

Anmerkung:
Lassen Sie den Code von `Graphic.java` unangetastet und verwenden Sie nur die vorgestellten Methoden dieser Klasse (dass das möglich ist, ist ein großer Vorteil der objektorientierten Programmierung).