Aufgabe 10

Bearbeitungs- und Abgabefrist: siehe moodle
20 Übungspunkte

Achten Sie bei dieser Aufgabe auf die exakte Einhaltung aller bisher vorgestellten technischen Regeln. Lesen Sie die Regeln ggf. in den Arbeitsblättern nach.

Aufgabe 10:

Verwenden Sie im Folgenden als Typ Point den Typ java.awt.Point

Hinweis: Vor dem Programmieren ist es sinnvoll, (i) die Kette der Datentransformationen nach Aufgabenteil c und (ii) die erwarteten Ergebnisse nach Aufgabenteil d, e zunächst auf Papier zu skizzieren.

a) Schreiben Sie die Klasse **pr1.a10.Data** Erzeugen Sie (in main) eine List<Integer> mit 75000 oder mehr zufälligen int-Werten im Bereich 0 <= i <= 800 und schreiben Sie
* die ersten 50000 Werte mit je 10 Werten je Zeile in die Datei ./data/A.txt
* die Werte ab der 25000 Zahl mit je 10 Werten je Zeile in die Datei ./data/B.txt.

b) Schreiben Sie das Interface **pr1.a10.PointFilter** mit der Methode
public boolean accept(java.awt.Point p) // Verwendung: siehe c)

c) Schreiben Sie die Klasse **pr1.a10.Convert** mit den Methoden
public static List<Integer> fileToList(String filename)
// verwendet jeweils zwei aufeinanderfolgende int-Werte als Koordinatenpaar:
public static Set<Point> intsToPoints(List<Integer> ints)
// erzeugt eine neues Set mit der gefilterten Punktmenge:
public static Set<Point> filter(Set<Point> points, PointFilter filter)
public static Set<Ellipse2d.Double> pointsToOvals(Set<Point> points)
public static Drawable ovalsToDrawable(Set<Point> points, Color color)
und
/** verwandelt die Point-Objekte in Ellipse2D.Double-Objekte (jedes sollte ein Punkt sichtbar, aber nicht zu groß darstellen), dann diese in ein Drawable Objekt und zeichnet dieses mithife des FunnyFirstPainters.
*/
public static void show(Set<Point> points, FunnyFirstPainter ffp)

d) Schreiben Sie die Klasse **pr1.a10.CircleFilter**, die PointFilter so implementiert, daß nur Point-Objekte akzeptiert werden, die in einem Kreis mit Radius 200 um den Punkt (300, 400) liegen.

Schreiben Sie die Klasse **pr1.a10.SquareFilter**, die PointFilter so implementiert, daß nur Point-Objekte akzeptiert werden, die in einem Quadrat mit Seitenlänge 300 um den Punkt (400, 500) liegen.

e) Schreiben Sie die Klasse **pr1.a10.Menge** mit den Methoden

```
/** liest die Datei ./data/A.txt verwandelt die Zahlen in Point-Objekte, diese in
Ellipse2D.Double-Objekte, dann diese in ein grünes Drawable Objekt a. Liest Datei ./data/B.txt
und erzeugt daraus  entsprechend das rote Drawable Objekt b. Zeichnet dann a und b mithife
eines FunnyFirstPainters in eine GUI.
*/
public static void showAandB()


/** ähnlich wie showAandB(), jedoch werden
  * die Punkte aus Datei A mit einem CircleFilter gefiltert bevor Sie in Ellipse...Objekte
verwandelt werden.
  * die Punkte aus Datei B mit einem SquareFilter gefiltert bevor Sie in Ellipse...Objekte
verwandelt werden.   Dann wieder gemeinsame Anzeige in einer GUI.
*/
public static void showCirclesAndSquares()


/** ähnlich wie showCirclesAndSquares(), aber zusätzlich:
    Aus dem Set mit den Circle-Punkten und dem Set mit den Square-Punkten wird die
Vereinigungsmenge gebildet und nur dies in ein Drawable-Objekt verwandelt, das dann in
einer GUI  gezeigt wird.
*/
public static void showVereingungmenge()


/** ähnlich wie showVereingungmenge(), aber anstelle der  Vereinigungsmenge wird aus
Circle und Square die Differenzmenge gebildet... dann ... GUI.
*/
public static void showDifferenzmenge()

/** ähnlich wie showVereingungmenge(), aber anstelle der  Vereinigungsmenge wird aus
Circle und Square die Schnittmenge gebildet... dann ... GUI.
*/
public static void show Schnittmenge()
```

f) Schreiben Sie die Klasse pr1.a10.**Test**

, in der Sie die Methoden aus e prüfen.