Prof. Dr. Stephan Kleuker
Hochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Programmierung 1

Wintersemester 2022/23
Aufgabenblatt 10

33. Aufgabe (4 Punkte, geschachtelte Schleifen, Fortsetzung letztes Aufgabenblatt)

Nutzen Sie die Klasse Messreihe, die als Objektvariablen den Messort und eine Sammlung von ganzzahligen Messwerten enthält, vom vorherigen Aufgabenblatt. Es gelten die gleichen Randbedingungen. Auf der Veranstaltungsseite finden Sie ein Projekt mit Testklassen, stellen Sie am Ende sicher, dass alle Tests laufen.

Hinweise: Natürlich dürfen Sie weitere Methoden zum Strukturieren Ihres Programms oder bei Code-Wiederholungen schreiben. Wieder ist es sinnvoll, die Lösung erst auf Papier zu entwickeln.

a) Schreiben Sie eine Methode paarSumme(), die ein neues Messreihen-Objekt zurückgibt, die die Summe der ersten beiden, des dritten und vierten, also jedes Wertepaares enthält. Ist die Anzahl der Werte ungerade, wird die letzte Zahl so übernommen, aus {4,3,2,4,3} soll z. B. {7,6,3} werden (4+3 = 7, 2+4 = 6, 3 alleine). Existiert die Liste nicht oder ist sie leer, enthält Ergebnis eine einelementige Liste mit dem Wert 0.

b) Schreiben Sie eine Methode gesamtsumme(), die die Summe aller Messwerte zurück gibt und dabei zum Rechnen ausschließlich die Methode paarSumme() aus a) nutzt.

c) Schreiben Sie eine Methode dreier(), die einen int-Wert übergeben bekommt und genau dann true liefert, wenn es drei Elemente an verschiedenen Positionen in der Liste gibt, die addiert diesen int-Wert ergeben. Z. B., für den Parameter-Wert 10 und die Liste {4,3,2,4,3} soll z. B. das Ergebnis true (4 + 2 + „andere 4“ = 10; auch 4 3 3 möglich) und die Liste {5,5,1,1,3} das Ergebnis false sein.

d) Schreiben Sie eine Methode werteNachHinten(), die einen int-Wert w übergeben bekommt und alle Vorkommen von w nach hinten in die Messwerte schiebt, die restliche Reihenfolge bleibt unverändert. Z. B., für den Parameter-Wert 4 und die Liste {4,3,2,4,3} soll z. B. das Ergebnis die Liste {3,2,3,4,4}, für den Wert 2 die Liste {4,3,4,3,2} und für den Wert 5 die unveränderte Liste enthalten. (Erinnerung: Die Methode remove() darf nicht genutzt werden, set() schon.)

34. Aufgabe (2 Punkte, equals und clone)

Von der Veranstaltungswebseite ist ein Projekt mit den Klassen Mitarbeit und Pairprogramming erhältlich, deren Anfang wie folgt aussieht.

```
public class Mitarbeit {
  private int minr;
  private String name;
```

```
public class Pairprogramming {
  private Mitarbeit mitglied1;
  private Mitarbeit mitglied2;
```

Die Klassen enthalten jeweils einen Konstruktor, get- und set-Methoden sowie leider falsche equals()- und clone()-Methoden. Korrigieren Sie diese Methoden mit den in der Veranstaltung vorgestellten Ansätzen und prüfen Sie ihre Ergebnisse mit den gegebenen Tests. Beachten Sie, dass alle Objektvariablen, deren Typ von Object erbt (also name, mitglied1, mitglied2), null-Werte enthalten dürfen. Zwei Pairprogramming-Objekte sollen nur gleich sein, wenn jeweils mitglied1 und mitglied2 übereinstimmen. Um bei Strings sicherzustellen, dass man eine Kopie mit gleichem Inhalt erhält, kann der Konstruktor genutzt werden, z. B. `new String(this.name)`.

35. Aufgabe (12 Punkte, konsequente Anwendung dynamischer Polymorphie)

In dieser Aufgabe sollen gefüllte Figuren und Figuren, aus denen andere Figuren ausgeschnitten sind, gezeichnet werden. Zur Zeichnung gefüllter Figuren gibt es im Interaktionsbrett nur die Möglichkeit die einzelnen Punkte der jeweiligen Figur zu zeichnen. Die Grundidee zum Darstellen komplexer, teilweise ausgefüllter Figuren wird in Abb. 1 skizziert. Gegeben ist eine Figur f, dann wird für jeden Punkt p, der zur Figur gehören könnte, mit f.innerhalb(p) geprüft, ob der Punkt in der Figur liegt (j) oder nicht (n). Ist der Punkt innerhalb, wird ein schwarzer Punkt an dieser Stelle gemalt. Sie können dabei Ihre Klassen Punkt, Kreis und Rechteck aus früheren Aufgaben in das auf der Veranstaltungsseite erhältliche Projekt kopieren und modifizieren.

Prof. Dr. Stephan Kleuker
Hochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Abhängig vom Zoom-Faktor könnte die Darstellung im Interaktionsbrett ein Raster haben, was ok ist. Bei sehr großem Zoom (>=5) sollte eine durchgehende Fläche dargestellt sein. Grundlage des Programms ist das auch von der Veranstaltungsseite erhältliche Interface Geoobjekt mit folgenden Methoden.

Method Summary	
void	**darstellen**(Interaktionsbrett ib) Methode zum Zeichen des Objekts auf einem Interaktionsbrettobjekt, dabei wird das Objekt komplett schwarz dargestellt.
boolean	**innerhalb**(Punkt p) Methode zur Berechnung, ob ein übergebener Punkt sich in diesem Objekt befindet, der Rand soll dabei zum Objekt gehören.
Punkt	**linksoben**() Methode zur Berechnung der linken oberen Ecke des Objektes, dabei kann kein Punkt des Objektes sich weiter links oder oberhalb dieses Punktes befinden.
Punkt	**rechtsunten**() Methode zur Berechnung der rechten unteren Ecke des Objektes, dabei kann kein Punkt des Objektes sich weiter rechts oder unterhalb dieses Punktes befinden.
void	**verschieben**(int dx, int dy) Methode zum Verschieben des Objektes, dabei wird das gesamte Objekt auf der x-Achse um dx Punkte und auf der y-Achse um dy-Punkte verschoben.

a) Erweitern Sie Ihre Klasse Rechteck so, dass sie das Interface Geoobjekt realisiert, dabei soll ein Rechteck gefüllt dargestellt werden, wie es in Abb. 3 links-oben in der Ecke zu sehen ist.

b) Erweitern Sie Ihre Klasse Kreis so, dass sie das Interface Geoobjekt realisiert. Um den ausgefüllten Kreis zu zeichnen wird der Ansatz aus Abb. 1 genutzt: Es wird für jeden Punkt in dem von den Punkten linksoben() und rechtsunten() umrandeten Rechteck geprüft, ob dieser innerhalb() liegt; ist das der Fall, wird der Punkt gezeichnet (n=nein,nicht, j=ja).

n	n	n	n	n	n
n	n	n	n	j	j
n	n	j	j	j	j
n	j	j	j	j	j
j	j	j	j	j	j
j	j	j	j	j	j

Abbildung 1: punktweise Darstellung

Um festzustellen, ob ein Punkt im Kreis liegt, kann der Satz des Pythagoras angewandt werden, wie es in Abb. 2 skizziert ist. Dazu wird der Abstand des zu untersuchenden Punktes auf der X-Achse (in der Abbildung a) und auf der Y-Achse (in der Abbildung b) vom Mittelpunkt des Kreises bestimmt. Gilt dann $a^2 + b^2 <= radius^2$, liegt der Punkt im Kreis (der Rand soll dazugehören).

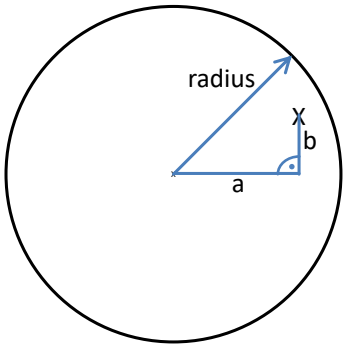

Abbildung 2: liegt Punkt X im Kreis?

Ein Beispiel für einen Kreis ist rechts neben dem Rechteck in Abb. 3 dargestellt.

c) Schreiben Sie eine Klasse Rahmen, die aus einem das Geoobjekt-Inferface realisierenden Objekt als Rahmen und mehreren daraus entfernten auch das Geoobjekt realisierenden Objekten besteht. Nutzen Sie dazu eine Methode `ausstanzen(Geoobjekt)` mit der aus dem Rahmen das übergebene Objekt

ausgestanzt wird. Es soll z. B. folgender Code möglich sein.

```
Rahmen ra = new Rahmen(new Rechteck(new Punkt(10,80),120,80));
ra.ausstanzen(new Kreis(new Punkt(15,85),20));
ra.ausstanzen(new Kreis(new Punkt(85,115),20));
```

Abb. 3 zeigt unter dem Rechteck z. B. einen Rahmen, der aus einem Rechteck-Objekt besteht, aus dem zwei Kreise entfernt wurden. Realisieren Sie die Klasse Rahmen so, dass sie selbst das Interface Geoobjekt realisiert. Sichtbar sind Punkte des Rahmens, die nicht auch in einem (oder mehreren) entfernten Objekten liegen. Der Ansatz mit innerhalb() aus Abb. 1 kann hier auch hilfreich sein.
Insgesamt kann damit ein Rahmen selbst wieder ein Rahmen in einem anderen Objekt sein. Die ineinander geschachtelten Rechtecke rechts neben den entfernten Kreisen in Abb. 3 sind so entstanden, indem aus einem äußeren Rahmen ein innerer Rahmen entfernt wurde.

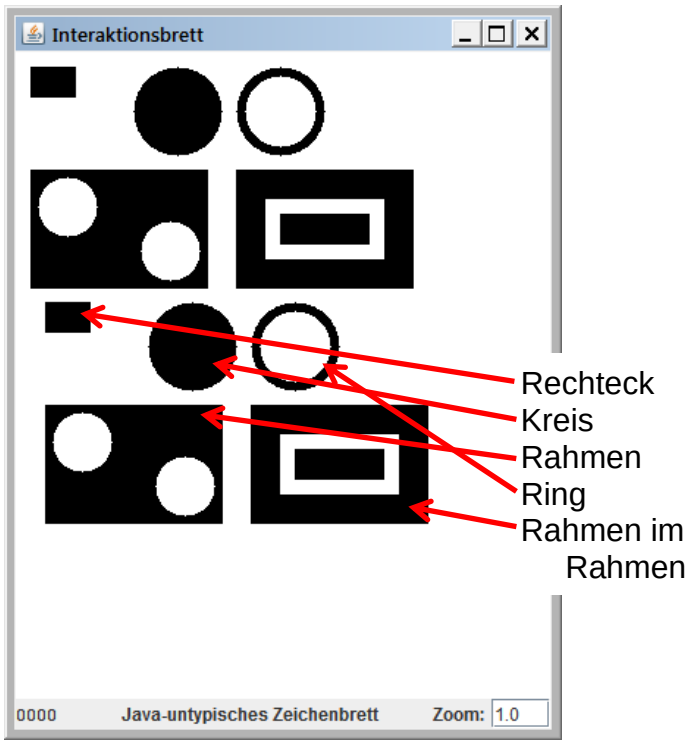

Abbildung 3: Interaktionsbrett mit Beispielfiguren

d) Lassen Sie eine Klasse Ring von der Klasse Rahmen erben, wobei bei einem Ring aus einem Kreis ein kleinerer Kreis mit gleichem Mittelpunkt entfernt wird.
Ein Konstruktor eines Ringes sollte wie folgt aussehen:

```
public Ring(Kreis aussen, int ringdicke)
```

Ein Ring-Beispiel ist in Abb. 3 neben den gefüllten Kreisen dargestellt.

Abbildung 4: robbi()

Zur Überprüfung Ihrer Ergebnisse ist die Klasse GeoSpielerei beigelegt, die das in Abb. 3 gezeigte Bild mit malMal() berechnet, bei dem mehrere Objekte auch einmal verschoben dargestellt werden. Die zweite Methode robbi() liefert das Bild aus Abb. 4.

Freiwillig: Da Code-Wiederholungen vermieden werden sollen, sind hier auch andere Ansätze sinnvoll. Überlegen Sie sich eine Lösungsvariante, in der es nicht zur Code-Wiederholung des Darstellungsverfahrens kommt.