RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
AACHEN

LEHR- UND FORSCHUNGSGEBIET INFORMATIK II
RWTH Aachen · D-52056 Aachen · GERMANY
http://programmierung.informatik.rwth-aachen.de

Prof. Dr. Jürgen Giesl
Darius Dlugosz, Thomas v. d. Maßen, Antje Nowack

Übung *Informatik I - Programmierung* - Blatt 9

Die Übungsblätter sollen in Gruppen von je 3 Studierenden bearbeitet werden. Diese 3 Studierenden müssen aus derselben Übungsgruppe kommen. Lösungen können bis zum 8. Januar, 17.45 h, in den Kasten für Ihre Übungsgruppe im Foyer in der Ahornstr. 55 eingeworfen werden.
Bitte vergessen Sie nicht, die **Nummer Ihrer Übungsgruppe** sowie die **Namen** und **Matrikelnummer** der Mitglieder Ihrer 3er-Gruppe auf Ihre Abgabe zu schreiben.

Wichtig: Alle Programme sind **sowohl** auf Papier in den Kasten einzuwerfen **als auch** elektronisch mit Hilfe des **Online-Systems** abzugeben.

Aufgabe 1 (9 Punkte)

In dieser Aufgabe soll die elektronische Girokontenverwaltung aus Aufgabe 4 des 5. Übungsblatts verändert werden. Die Klasse *Konto* soll nun in der Lage sein, beliebig viele Buchungen speichern zu können. Dazu ist es notwendig, die vorhandene Implementierung mit der Array-Verwaltung durch eine **doppelt-verkette** lineare Liste zu ersetzen. Beachten Sie dabei die folgenden Punkte:

- Definieren Sie für die Klasse *Konto* geeignete Konstruktoren die es erlauben, ein leeres aber initialisiertes Konto sowie ein Konto mit einer Belegung der Attribute zu instantiieren.
- Implementieren Sie die Verwaltung der Buchungen durch eine doppelt-verkette lineare Liste. Durch die doppelte Verkettung soll es möglich sein, in beide Richtungen durch die Liste zu navigieren. Dies bedeutet, dass jedes Listenelement einen Verweis auf seinen Nachfolger und auf seinen Vorgänger besitzt. Die Realisierung der Buchungsliste sollte in einer eigenen Klasse - analog zur Vorlesung - implementiert werden. Überlegen Sie sich Operationen, die für den nächsten Punkt und somit zur Verwaltung der Buchungsliste benötigt werden. Sie können den Quellcode aus dem Lösungsvorschlag der Aufgabe 4 des 5. Übungsblatts verwenden.
- Passen Sie die folgenden Methoden der Klasse *Konto* aus Aufgabe 4b des 5. Übungsblatts entsprechend an:
  - `public void buchungHinzufuegen(Buchung b)`
  - `public void druckeBuchungsliste()`
    Diese Methode soll die Buchungsliste vorwärts und rückwärts ausgeben!
  - `public double berechneSaldo()`

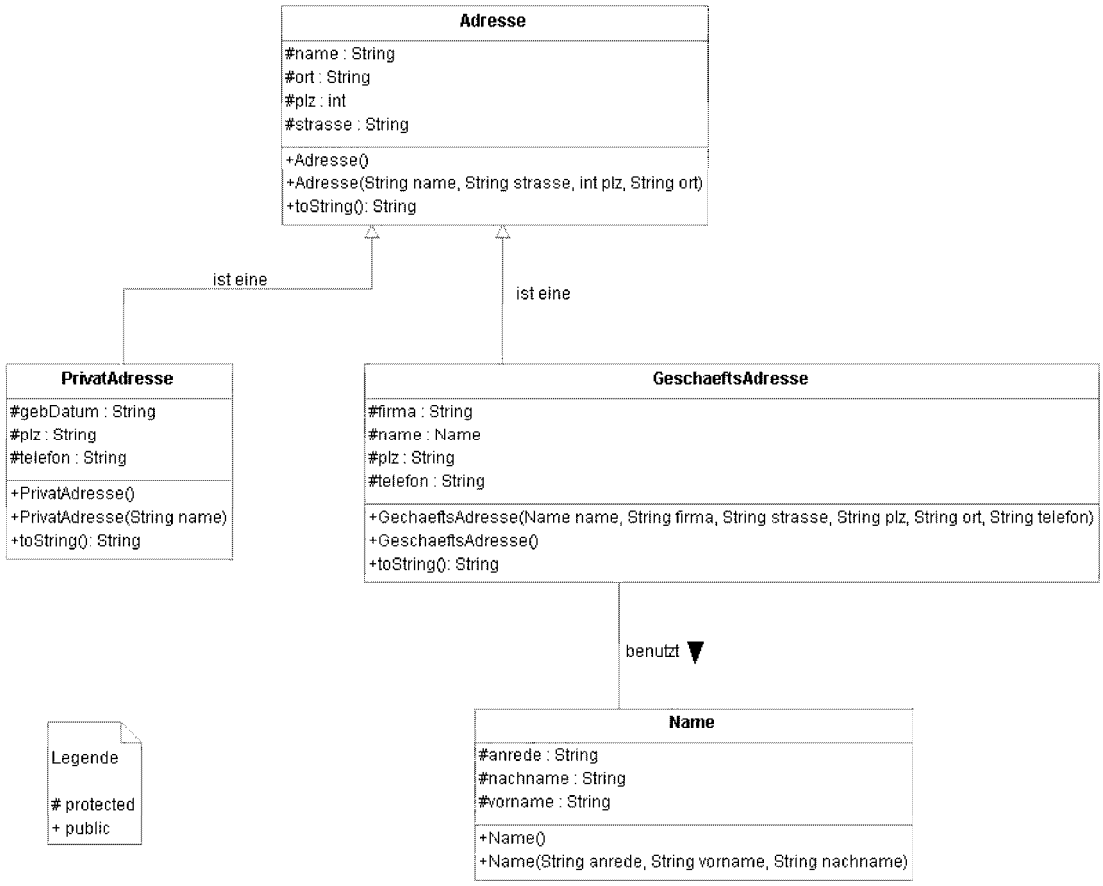
Hinweis: Diese Methoden können selbstverständlich Methoden der Klasse, welche die Buchungsliste realisiert, aufrufen. Beachten Sie, dass eine neue Buchung direkt an der richtigen Position (d. h. die chronologische Ordnung soll erhalten bleiben) eingefügt werden soll.

- Beachten Sie, dass Sie keine Änderungen an den Klassen *Buchung* oder *Datum* durchführen sollen.

Realisieren Sie alle Zugriffe auf die Liste mittels Rekursion, d.h. ohne Verwendung von Schleifen oder Sprüngen.

Aufgabe 2 (4 Punkte)

Gegeben sei das folgende Klassendiagramm:



Ebenso ist eine Klasse *Adressentest* gegeben:

```
1: public class Adressentest {
2:
```

```
 3:   public static void main (String[] args) {
 4:      Adresse adr = new Adresse("Muster", "Rheinstrasse 4", 52060, "Aachen");
 5:      PrivatAdresse padr = new PrivatAdresse();
 6:      GeschaeftsAdresse gadr = new GeschaeftsAdresse();
 7:
 8:      padr = gadr;
 9:      padr = (Adresse)gadr;
10:      gadr = adr;
11:      padr.name = "Peter Muster";
12:      padr.plz = "52074";
13:      padr.gebDatum = "13.4.1968";
14:
15:      adr = padr;
16:      gadr.telefon = "+49 241 123456";
17:      System.out.println("Name: " + adr.name);
18:      System.out.println("Telefon: " + adr.telefon);
19:
20:      Adresse a = new GeschaeftsAdresse();
21:      adr.telefon = "0241-10275";
22:      gadr.plz = "D-52074";
23:      gadr.name = "Simon Schmidt";
24:      padr.name = "Simon Schmidt";
25:      a.plz = "D-52074";
26:      a.name = new Name("Herr", "Klaus", "Wienert");
27:      gadr.name = new Name("Herr", "Klaus", "Wienert");
28:   }
29: }
```

Geben Sie für die Anweisungen der Zeilen 4-27 in der main-Methode an, ob diese gültig sind, d.h. ob sie ausführbar sind oder ob sie zu einem Fehler führen würden. Falls eine Anweisung nicht gültig ist, geben Sie den Grund dafür an. Sollte eine Anweisung gültig sein, so erläutern Sie, auf welches Attribut bzw. auf welche Methode welcher Klasse zugegriffen wird.

Aufgabe 3 (3 + 6 + 2 + 2 Punkte)

Ein Graphikeditor soll die folgenden geometrischen Figuren darstellen können:

- Rechtecke
- Kreise
- Dreiecke
- Quadrate
- Ellipsen

Alle Figuren besitzen einen Flächeninhalt und einen Umfang. Ebenso besitzen alle Figuren eine x- und eine y-Koordinate sowie eine Linien- und eine Füllfarbe.

a) Modellieren Sie die oben angegebenen Klassen und fassen sie gleiche Funktionalität in sinnvollen Oberklassen zusammen, d.h. so viele Attribute und Mathoden wie möglich sollen vererbt werden. Zeichnen Sie ein Klassendiagramm gemäß Aufgabe 2, welches die benötigten Klassen mit ihren Attributen und Methodensignaturen zeigt sowie die Vererbungsbeziehungen zwischen den Klassen darstellt. Alle Klassen sollen u.a. die folgenden Methoden bereitstellen:

   - `public double berechneFlaecheninhalt()`
   - `public double berechneUmfang()`

b) Realisieren Sie die von Ihnen modellierten Klassen in Java. Verwenden Sie dabei die Konzepte, die im Kapitel II.4 der Vorlesung vorgestellt worden sind. Entwerfen Sie insbesondere geeignete Konstruktoren, um die Attribute der jeweiligen Klasse mit beliebig vorgegebenen Werten zu initialisieren. Verwenden Sie dabei soweit wie möglich bereits existierende Konstruktoren der Oberklassen unter Benutzung von `super(...)`. Sollte es in einer Oberklasse nicht möglich sein, die Berechnungen von Umfang und Flächeninhalt durchzuführen, da noch zu wenige Informationen über die geometrische Figur vorhanden sind, so soll der Benutzer nach den Werten für den Flächeninhalt und Umfang gefragt werden.

c) Implementieren Sie eine Methode in einem Hauptprogramm, welche als Parameter ein Array erhält, welches beliebige geometrische Figuren enthält. Die Methode soll die Anzahl der jeweiligen geometrischen Figuren (d.h. die Anzahl der Rechtecke, der Quadrate, ...), die in dem Array gespeichert sind sowie die Summe aller Flächen berechnen und ausgeben.

d) Erzeugen Sie für alle Klassen eine gemeinsame Schnittstellendokumentation mit `javadoc`.

Hinweis: Die folgenden (Näherungs-)Formeln können zur Berechnung von Flächeninhalt und Umfang verwendet werden:

- Fläche eines allgemeinen Dreiecks: $A = \sqrt{s * (s-a) * (s-b) * (s-c)}$, wobei s für den halben Umfang steht
- Fläche einer Ellipse: $A = \pi * a * b$, wobei a und b die Längen der beiden Halbachsen sind
- Umfang einer Ellipse: $U \approx \pi * (3 * (a+b) - \sqrt{(a+3*b) * (3*a+b)})$