Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik III
Prof. Dr. A.B. Cremers

Aufgabenblatt 6

— Veröffentlicht am 10.06.2010 —

Abgabe bis 15.06.2010, 13:00 per SVN

Aufgabe 1 *(Typen, Subtypen, Nachrichten - 3 Punkte)*

Überlegen Sie sich für jede Aussage ein Argument, mit dem Sie eindeutig bestimmen können, ob die Aussage wahr oder falsch ist. Für jede richtig angekreuzte Antwort werden 0,5 Punkte angerechnet, für jede falsch angekreuzte Antwort 0,5 Punkte abgezogen. Nicht angekreuzte Antworten geben keinen Punktabzug.

wahr	falsch	
		Der genaue Typ eines Objektes kann vom Compiler aus den Deklarationen im Programmtext hergeleitet werden.
		Die Aufrufsignatur ergibt sich aus den statischen Typen der Aufrufargumente
		Der statische Typ der linken Seite einer Zuweisung darf kein Untertyp des Typs der rechten Seite sein.
		Eine Typkonversion vom Untertyp zum Obertyp wird als *Erweiterung* (*widening*) bezeichnet.
		Eine Typ-Erweiterung ist nur durch einen expliziten 'cast'-Operator möglich.
		Seien $U_1 \leq T_1$, $U_2 \leq T_2$ die einzigen Subtypbeziehungen zwischen den Typen T_1, T_2, U_1, U_2. Seien ferner $m(T_1, T_2)$ und $m(U_1, T_2)$ die Kandidatensignaturen für einen Aufruf mit der Signatur $m(U_1, U_2)$. Dann ist der Aufruf mehrdeutig.

Aufgabe 2 *(Overloading - Overriding - 8,5 Punkte)*

a) Erklären Sie den Unterschied zwischen *Überladen (Overloading)*, *Überschreiben (Overriding)* und *Verdecken (Shadowing / Hiding)*.

b) Beschreiben Sie die Schritte die beim Zugriff auf eine Instanzmethode ablaufen. Erläutern Sie welche davon das Überladen, Überschreiben bzw. Verdecken implementieren.

c) Was ist beim Zugriff auf eine Klassenmethode anders?

Aufgabe 3 *(ADT - 10 Punkte)*

In dieser Aufgabe soll ein abstrakter Datentyp programmiert werden, welcher komplexe Zahlen repräsentiert. Die komplexen Zahlen sind Ihnen sicherlich aus Ihren Mathematikvorlesungen bekannt. Falls dies bei Ihnen nicht der Fall sein sollte, finden Sie alles Nötige in dem Buch *Analysis 1* von Otto Forster.

Implementieren Sie einen abstrakten Datentyp *Complex*, welcher die Operationen Addition, Subtraktion, Multiplikation, Division, Konjugation und Betrag auf komplexen Zahlen unterstützt. Das folgende Testprogramm sollte mit Ihrem ADT funktionieren:

```
public class Complex {

/* Implementieren Sie den ADT hier! */

public static void main(String[] args) {
    Complex c1 = new Complex(2.4, -3.1);
```

```
    Complex c2 = new Complex(-1.0, -12.4);
    Complex c3 = new Complex(3.1415); // reelle Zahl
    Complex c4 = new Complex();  // Null

    System.out.println("Der Realteil von " + c1 + " ist " + c1.getReal());
    System.out.println("Der Imaginärteil von " + c1 + " ist " + c1.getImaginary());

    System.out.println("Die Summe von " + c1 + " und " + c2 + " ist " + c1.add(c2));
    System.out.println("Die Differenz von " + c1 + " und " + c3 + " ist " + c1.subtract(c3));
    System.out.println("Das Produkt von " + c1 + " und " + c2 + " ist " + c1.multiply(c2));
    System.out.println("Der Quotient von " + c1 + " und " + c2 + " ist " + c1.divide(c2));

    if (c1.equals(c4))
      System.out.println(c1 + " ist gleich " + c4);
    else
      System.out.println(c1 + " ist nicht gleich " + c4);

    System.out.println("Die zu " + c1 + " konjugiert komplexe Zahl lautet " + c1.conjugate());
    System.out.println("Der Absolutbetrag von " + c1 + " ist " + c1.abs());
  }
}
```

Aufgabe 4 *(Produktkatalog - 9 Punkte)*

Gegeben sei die folgende Schnittstelle, die ein Produkt aus einem Katalog repräsentiert:

```
public interface Product {
  String getName();             // Produktbezeichnung
  String getDescription();      // Produktbeschreibung
  double getPrice();            // Netto-Preis
  double getPrice(float tax) ; // Netto-Preis plus 'tax' Prozent Mehrwertsteuer
  public String toString() ;    // Textdarstellung aller obigen Informationen
                                // incl. Preis mit 19% Mehrwertsteuer.
}
```

a) Als konkrete Produkte sollen Sie Kleidung und Bücher darstellen. Kleidung ist durch eine Größe und das Material gekennzeichnet. Bücher sind durch einen Autor und einen Titel gekennzeichnet (jeweils als `String`).

Schreiben Sie Klassen, die Kleidung (`Clothing`) und Bücher (`Book`) darstellen und jeweils das Produkt-Interface implementieren. Alle Attribute sollen im jeweiligen Konstruktor mit Werten belegt werden. Die Methode `toString()` der jeweiligen Produktart soll auch die Werte der zusätzlichen Attribute in den Rückgabe-String integrieren.

Bevor Sie sich in die Implementierung stürzen, überlegen Sie, wie Sie Ihre Klassen mit den in der Vorlesung vorgestellten Mitteln möglichst redundanzfrei realisieren können.

b) Schreiben Sie eine Klasse `Catalogue`, deren Instanzen jeweils einen Produktkatalog darstellen, das heisst, jede `Catalogue`-Instanz verwaltet eine Menge von beliebigen Produkten. (Hinweis: Eine Menge von Produkten können Sie mit Hilfe eines `HashSet<Product> products;` verwalten (siehe *java.util.HashSet*).

Beim Aufruf der `printMe()` Methode einer Katalog-Instanz sollen String-Repräsentationen aller enthaltenen Produkte mit Hilfe der jeweils entsprechenden `toString()`-Methode auf der Kommandozeile ausgedruckt werden.

c) Fügen Sie zur Klasse `Catalogue` eine `main`-Methode hinzu, die die folgenden Objekte erzeugt, zu einem Katalog hinzugefügt und dann den Katalog ausdruckt:

- ein `Book` mit dem Titel *Freibier für alle*, dem Autor *Westerwelle et al.*, der Beschreibung *Unfreiwillige Satire in der deutschen Politik* und einem Preis von *7,99 EUR*.
- ein `Clothing` mit dem Namen *Hemd*, der Beschreibung *Herren-Hemd*, einem Preis von *33,61 EUR*, der Größe *48* und dem Material *Baumwolle*.