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

Aufgabenblatt 10

— Veröffentlicht am 06.07.2010 —

Abgabe bis 13.07.2010, 13:00 per SVN

Aufgabe 1 *(Generics, Listen - 2 Punkte)*

Überlegen Sie sich für jede Aussage, ob sie wahr oder falsch ist und kreuzen Sie das entsprechende Feld an. Jede richtige Antwort ergibt 0,5 Punkte, jede Falsche -0,5 Punkte und jede Fehlende 0 Punkte. Also: Besser nicht ankreuzen, statt falsch ankreuzen!

wahr	falsch	
		Ein Datentyp heißt generisch, wenn er hinsichtlich der Anzahl seiner Elemente parametrisiert ist.
		Ein Algorithmus heißt generisch, wenn er unabhängig von einem Datentyp programmiert werden kann.
		`Set<Object>` ist Obertyp von `Set<String>`.
		Anders als in einem Array lassen sich in einer Liste Objekte verschiedener Typen speichern.

Aufgabe 2 *(Listen - 15 Punkte)*

a) Formen Sie die in der Vorlesung vorgestellte Implementierung der Klassen `TNode` und `TList` jeweils zu *generischen* Versionen mit Typparameter T um (d.h. `TList<T>` und `TNode<T>`)!

b) Erweitern Sie `TList<T>` um die Methoden

- `T get(int i)`, die das i-te Element der Liste zurückgibt,
- `void delete(int i)`, die das i-te Element aus der Liste entfernt,
- `void insert(T elem, int i)`, die das Element `elem` an der i-ten Stelle der Liste einfügt.

Analog zur Array-Indizierung sei hierbei $i = 0$ die Position des ersten Listenelements. Behandeln Sie jeweils die Ausnahme, dass der Index i die Länge der Liste übersteigt!

c) Erweitern Sie die Klasse `TList<T>` um die folgende Methoden:

- `public void concat(List<T> a)` – konkateniert Listen *destruktiv*, d.h. hängt an das Ende der `this`-Liste die Liste `a` an,
- `public List<T> append(List<T> a)` – konkateniert Listen konstruktiv, d.h. es wird eine neue Liste zurückgegeben.

Aufgabe 3 *(Matrizenmultiplikation - 11 Punkte)*

a) Implementieren Sie eine Klasse `DoubleMatrix`, die Matrizen allgemeiner Dimension von Zahlen vom Typ `double` repräsentiert. Es sollen also nicht nur quadratische Matrizen behandelt werden können, sondern allgemeine Matrizen, die lineare Abbildungen von $\mathbb{R}^m \to \mathbb{R}^n$ darstellen (wobei $\mathbb{R}$ durch `double` approximiert wird und n und m positive Zahlen vom Typ `int` sind). Implementieren Sie eine Funktion `matrixMult` zur Durchführung der Matrizenmultiplikation!

Kapseln Sie die Zugriffe auf Zeilen, Spalten und Elemente der Matrix durch passende Zugriffsmethoden.

b) Behandeln Sie die Ausnahme, dass die Operation wegen inkompatibler Dimensionen der Matrizen nicht durchgeführt werden kann! Geben Sie als Teil der Ausnahmebehandlung eine Fehlermeldung auf `System.err` aus! Dokumentieren Sie ihren Quellcode mit `javadoc`!

c) Implementieren Sie eine Klasse `GenericMatrix<T extends Number>`, welche dieselbe Funktionalität besitzt wie die Klasse `DoubleMatrix`, jedoch den Datentyp Matrix generisch für Elemente von beliebigen Zahlentypen realisiert.