Fachbereich Informatik und Kommunikation
Prof. Dr. Marcel Luis

Westfälische Hochschule

23. Mai 2012

Aufgabenblatt 7 zu Objektorientierte Programmierung

Aufgabe 7.1

Endliche Folgen ganzer Zahlen könnte man durch Objekte des Typs `List<Integer>` oder `int[]` repräsentieren. Aber wie sieht es mit *unendlichen* Folgen aus?

Eine elegante Möglichkeit, endliche *und* unendliche Folgen gleichermaßen zu repräsentieren, besteht in Klassen, die eine Schnittstelle `Zahlenfolge` implementieren. `Zahlenfolge` soll folgende Methoden deklarieren:

- Eine Methode `boolean hatNaechstes()`, die genau dann `true` liefert, wenn die Zahlenfolge noch ein weiteres Element enthält.
- Eine Methode `int naechstes() throws NoSuchElementException`, die das nächste Element liefert oder eine Ausnahme wirft, wenn die Folge kein Element mehr enthält.

(Damit ist die Schnittstelle `Zahlenfolge` vergleichbar der Schnittstelle `Enumeration`, jedoch eingeschränkt auf Werte des Typs `int`.)

Definieren Sie die Schnittstelle `Zahlenfolge` und realisieren Sie folgende Klassen, die diese Schnittstelle implementieren:

- Eine Klasse `EndlicheFolge` mit einem Konstruktor `EndlicheFolge(int[])`. Ein Objekt dieser Klasse repräsentiert eine endliche Folge, deren Werte beim Erzeugen explizit angegeben werden.
- Eine Klasse `FibonacciFolge` mit einem Konstruktor `FibonacciFolge()`. Ein Objekt dieser Klasse repräsentiert die Folge der Fibonacci-Zahlen.
- Eine Klasse `Mischfolge` mit einem Konstruktor `Mischfolge(Zahlenfolge, Zahlenfolge)`. Ein Objekt dieser Klasse repräsentiert die Folge, die aus den gemeinsamen Werten zweier Folgen besteht. Sind diese beiden Folgen sortiert, ist die Mischfolge ebenfalls sortiert. (Ansonsten kann die Mischfolge die Elemente in einer beliebigen Reihenfolge liefern. Dies bedeutet, Sie können sich bei der Realisierung auf den Fall der sortierten Folgen konzentrieren und müssen lediglich dafür sorgen, dass im anderen Fall kein Element „unter den Tisch“ fällt.)

  Das „Dilemma“ bei dieser Klasse ist, dass Sie für die Methode `naechstes` nie im Vorhinein wissen, in welcher Folge sich das nächstkleinste Element befindet. Also werden Sie wohl auf

beide Folgen zugreifen müssen. Was passiert aber mit dem größeren der beiden Elemente?

Statt hier eine spezielle „Das-Element-merke-ich-mir-für-später-Strategie“ zu implementieren, könnten Sie eine Strategie anwenden, die bei Eingabeströmen, aus denen man wie bei unseren Zahlenfolgen ebenfalls nur „vorwärts“ lesen kann, angewandt wird. Dort gibt es Klassen `PushBackInputStream` und `PushBackReader`, die ermöglichen, Bytes oder Zeichen in den Eingabestrom zurückzuschreiben. Realisieren Sie deshalb ...

- Eine Klasse `PushBackFolge` mit einem Konstruktor `PushBackFolge(Zahlenfolge)`. Ein Objekt dieser Klasse basiert auf einer Zahlenfolge und ergänzt diese um die Fähigkeit, Werte „zurückzuschreiben“. Dazu dient die Methode `schreibeZurueck(int)`. Wird diese Methode mehrfach hintereinander aufgerufen, liefert die nächste Anwendung der Methode `naechstes` das zuerst zurückgeschriebene Element.

- Eine Klasse `EindeutigeFolge` mit einem Konstruktor `EindeutigeFolge(Zahlenfolge)`. Ein Objekt dieser Klasse basiert auf einer Zahlenfolge und repräsentiert deren Werte ohne doppelte Elemente. Es wird davon ausgegangen, dass die übergebene Folge sortiert ist.

Zum Testen und für die genaue Semantik der Methoden sind Ihnen für jede der angegebenen Klassen Testklassen vorgegeben.

Hinweise

- Verwenden Sie nur den Vorlesungsstoff bis einschließlich Kapitel 10.

- Denken Sie an die ausreichende Dokumentation und Kommentierung Ihrer Lösung. Beachten Sie die unterschiedliche Bedeutung der *externen Dokumentation* `/** ... */` *vor* einer Klasse oder Methode und des *Implementierungskommentars* `/* ... */` *innerhalb* einer Methode. Die externe Dokumentation sagt, *was* eine Klasse oder Methode leistet, der Implementierungskommentar hilft zu verstehen, *wie* es gemacht wird. Verwenden Sie Implementierungskommentare vor allem, um den Berechnungsablauf verständlich zu machen.

- Erzeugen Sie die HTML-Dokumentation Ihrer Klasse und überzeugen Sie sich, ob Ihre externe Dokumentation sinnvoll und ohne Kenntnis des Quellcodes der Klasse hilfreich ist.

- Laden Sie als Lösung dieser Aufgabe die Schnittstelle `Zahlenfolge` und alle von Ihnen realisierten Klassen zu Moodle hoch.

- Im Veranstaltungskalender finden Sie die Termine, an denen diese Lösung im Praktikum besprochen wird. Bringen Sie zu diesen Terminen bitte die Auswertung Ihrer Lösung mit, entweder ausgedruckt oder unmittelbar auf Ihrem Rechner verfügbar.