Prof. Dr. Volker Linnemann
Dr. Sven Groppe
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 30.11.2007

Vorlesung „Programmieren“

Wintersemester 2007/2008

6. Übungsblatt

1. Programmierung einer Keller-Klasse (Programmieraufgabe) (10 Punkte)

Implementieren Sie eine Klasse, die die Funktionalität eines *endlich beschränkten* Kellers zur Verfügung stellt. Die einzelnen Elemente des Kellers sollen `int`-Werte beinhalten. Zur Speicherung der Elemente des Kellers soll ein `int[]`-Feld verwendet werden, welches im Konstruktor mit einer in einem Parameter übergebenen Größe erzeugt wird.

Aufgaben:

a) Implementieren Sie den Konstruktor `public Stack(int length)` der Klasse `Stack`, der ein `int[]`-Feld der Größe `length` erzeugt, welches zur Speicherung der Elemente des Kellers verwendet wird. Eventuell sind weitere Initialisierungen hier notwendig. (2 Punkte)

b) Implementieren Sie die Methode `public boolean isEmpty()` der Klasse `Stack`, die `true` zurückgibt, falls der Keller leer ist. Ansonsten gibt diese Methode `false` zurück. (2 Punkte)

c) Implementieren Sie die Methode `public boolean isFull()` der Klasse `Stack`, die `true` zurückgibt, falls der endlich beschränkte Keller voll ist. Ansonsten gibt diese Methode `false` zurück. (2 Punkte)

d) Implementieren Sie die Methode `public int pop()` der Klasse `Stack`, die das oberste Element vom Keller löscht und als Ergebnis der Methode zurückgibt. (2 Punkte)

e) Implementieren Sie die Methode `public void push(int element)` der Klasse `Stack`, die `element` als das oberste Element auf den Keller legt. (2 Punkte)

Bemerkungen:

- Dieses ist eine Programmieraufgabe. Bitte geben Sie die Lösung *sowohl* auf **Papier** *als auch* den Java-Quelltext in elektronischer Form per **Email** an ihren Tutor ab.

2. Überprüfung eines Klammergebirges mit Klammern unterschiedlichen Typs (Programmieraufgabe) (10 Punkte)

In dieser Aufgabe sollen 3 unterschiedliche Typen von Klammern unterschieden werden: Die öffnende Klammer „`(`“ mit der dazugehörigen schließenden Klammer „`)`“, die öffnende Klammer „`{`“ mit der dazugehörigen schließenden Klammer „`}`“ und die öffnende Klammer „`[`“ mit der dazugehörigen schließenden Klammer „`]`“.

Ein korrektes Klammergebirge ist ein String mit beliebigem Inhalt (auch der leere String), der die oben angegebenen öffnenden („`(`“,„`{`“ und „`[`“) und schließenden („`)`“, „`}`“ und „`]`“) Klammern *nicht* enthält. Ein korrektes Klammergebirge ist auch *K1 K2*, wobei *K1* und *K2* korrekte Klammergebirge repräsentieren. Ein korrektes Klammergebirge ist weiterhin *Ö K S*, wobei *Ö* eine öffnende Klammer, *K* ein korrektes Klammergebirge und *S* eine zu *Ö* dazugehörige schließende Klammer repräsentiert. Korrekte Klammergebirge sind demnach beliebige Zeichenketten, die die öffnenden und schließenden Klammern in richtiger Klammerreihenfolge enthalten müssen. Klammern können auch beliebig ineinander geschachtelt sein.

Beispiele:

„`jksdi`“, „`h(k(())kl)ll`“, „`({[[]]})`“ und „`{ui[hkd]jhh[kk]}()`“ sind korrekte Klammergebirge, aber „`dhjfsdkj(`“, „`([]}`“ und „`{}(((((())))`“ sind *keine* korrekten Klammergebirge.

In dieser Aufgabe soll nun eine Methode entwickelt werden, die ein Klammergebirge auf Korrektheit überprüft und dabei einen endlich beschränkten Keller verwendet.

Aufgabe:

a) Für die Lösung dieser Aufgabe soll ein endlich beschränkter Keller verwendet werden. Für wie viele Elemente muss der Keller mindestens Platz reservieren in Abhängigkeit von der Länge des zu überprüfenden Klammergebirges, damit immer der Platz ausreicht, um das Klammergebirge auf Korrektheit zu überprüfen? Bitte erklären Sie! (1 Punkt)

b) Implementieren Sie eine Klasse `bracketMountain`, die eine Methode `public boolean check(String s)` besitzt. Die Methode `public boolean check(String s)` soll dabei `true` zurückgeben, falls `s` ein korrektes Klammergebirge beinhaltet, ansonsten soll die Methode `false` zurückgeben. (9 Punkte)

Bemerkungen:

- Bei dieser Aufgabe muss die Klasse `Stack` der ersten Aufgabe auf diesem Übungsblatt verwendet werden.
- Seien `s`, `s1` und `s2` Variablen vom Typ `String`. Die folgenden Klassen und Methoden können als bekannt vorausgesetzt und zur Lösung der Aufgabe verwendet werden: Die Methode `s.substring(i,i+1)` gibt einen String zurück, der nur das Zeichen an der Position i von s enthält. Die Positionen werden dabei wie bei Feldern beginnend bei 0 durchnummeriert. `s1.compareTo(s2)` gibt `0` zurück, falls der Inhalt von `s1` genau dem Inhalt von `s2` entspricht. `s.length()` gibt die Länge (d.h. Anzahl Zeichen) des Strings `s` zurück.
- Dieses ist eine Programmieraufgabe. Bitte geben Sie die Lösung *sowohl* auf **Papier** *als auch* den Java-Quelltext in elektronischer Form per **Email** an ihren Tutor ab.

Bemerkungen:

- Jede Seite soll oben rechts den Namen der Abgebenden und die Übungsgruppennummer (wichtig!) enthalten.
- Lösungen für die Übungsaufgaben sind (in der Regel) zu zweit abzugeben.
- Kommentieren Sie Ihre Lösungen! Besteht eine Lösung aus mehreren Zetteln, so sind diese zusammen zu heften. Bitte keine Hüllen, Mappen, o.ä..
- Bitte schicken Sie *Programmieraufgaben zusätzlich zur Abgabe auf Papier in elektronischer Form per Email* an ihren jeweiligen Tutor.
- Wenn nicht ausdrücklich anders in der Aufgabenstellung erwähnt, sollen Standard Java-API Klassen und Methoden *nicht* verwendet werden, sondern die Funktionalitäten von Grund auf neu entwickelt werden. Es geht bei diesen Aufgaben nicht um die Erlernung der Verwendung der Standard Java-API, sondern um die Erlernung der dahinter liegenden Methoden und Konzepte.
- Kommentieren Sie ihren Quelltext bei Programmieraufgaben. Dabei sollen keine Trivialitäten kommentiert werden, also bitte keine Kommentare wie

  ```
  x=5; // Wir weisen nun der Variablen x den Wert 5 zu
  ```

  sondern sinnvolle Kommentare, die Ideen des Quelltextabschnittes beschreiben oder auf Unteraufgaben (z. B. a), b), …) hinweisen.
- **Hinreichende Bedingung für die Zulassung zur Klausur:** 50% der erreichbaren Punkte bei jedem Übungszettel (bis auf zwei) und einmaliges Vorrechnen in der Übung
- **Zertifikatskriterium:** Das Bestehen der Klausur am Ende des Semesters

Abgabetermin: Freitag, 7.12.2007, nach der Vorlesung