TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
Prof. Dr. Helmut Seidl

SS 2008
Aufgabenblatt 9
16. Juni 2008

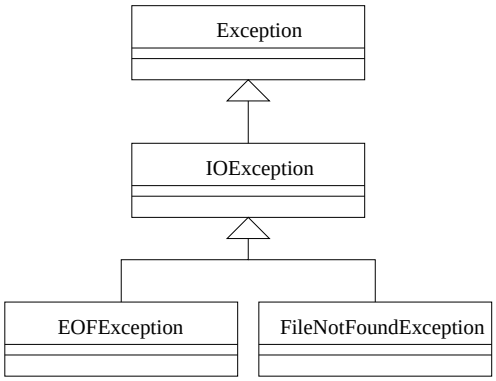
Abgabe: 24.-26. Juni 2008 beim jeweiligen Tutor

Praktikum Grundlagen der Programmierung

Themen: Exceptions, Ein- und Ausgabe

Aufgabe 42 (Ü) Ausnahmen

Gegeben sei folgender Ausschnitt aus der Klassenhierarchie von Ausnahmen in Java und folgende Java–Implementierung der Klasse `ExceptionTest`:




```
import java.io.*;
public class ExceptionTest {
   public static void main (String[] args) {
      try {
        ... // Inneres des try-Blockes
      }
      catch (EOFException e) {
         System.out.println("EOFException");
      }
      catch (IOException e) {
         System.out.println("IOException");
      }
      catch (Exception e) {
         System.out.println("Exception");
      }
      System.out.println("ENDE");
   }
}
```

a) An der durch „`...`“ gekennzeichneten Stelle im `try`–Block stehe ein Programmstück, durch das Ausnahmen vom Typ `IOException`, `EOFException` oder `FileNotFoundException` ausgelöst werden können.
   Was wird bei Ausführung der `main`–Methode ausgedruckt, falls dabei im `try`–Block
   i) als erstes eine Ausnahme vom Typ `EOFException` ausgelöst wird,
   ii) als erstes eine Ausnahme vom Typ `FileNotFoundException` ausgelöst wird, oder
   iii) gar keine Ausnahme ausgelöst wird?

b) Was wird bei Ausführung der `main`–Methode ausgedruckt, falls dabei im `try`–Block als erste Ausnahme eine Division durch `0` auftritt?

Aufgabe 43 (Ü) IO und Exceptions

Ziel dieser Aufgabe ist es, eine statische Methode **`int`** `berechneWert(String datei)` zu schreiben, die den Wert eines in einer Datei gespeicherten arithmetischen Ausdrucks berechnet. Dabei beschränken wir uns auf arithmetische Ausdrücke, in denen lediglich **natürliche Zahlen**

und **Addition** auftreten. Als Beispiel soll ein Aufruf der geforderten Methode für den Ausdruck `21 + 2 + 19` die Zahl `42` zurück liefern.

Dabei ist folgendes zu beachten:

- Ist das Format der Datei falsch, so soll bei einem Aufruf der Methode `berechneWert()` eine `FalschesFormatException` geworfen werden, die die Information enthält, beim Lesen des **wie vielten Wortes** ein Fehler aufgetreten ist.
- Existiert keine Datei namens `datei`, so soll eine `DateiNichtVorhandenException` geworfen werden.

Zur Lösung dieser Aufgabe können Sie die nachfolgend beschriebene Klasse `DateiLeser` verwenden, mit der eine Datei Wort für Wort gelesen werden kann. Die Klasse `DateiLeser` ist gegeben und muss **nicht** von Ihnen implementiert werden.

- Der Konstruktor `DateiLeser(String datei)` erzeugt ein `DateiLeser`-Objekt für die Datei `datei`. Existiert eine solche Datei nicht, so wird eine `DateiNichtVorhandenException` geworfen.
- Um Wort für Wort lesen zu können, wird die Methode `String leseNaechstesWort()` zur Verfügung gestellt. Diese liefert beim *i*-ten Aufruf das *i*-te Wort als String zurück. Existiert kein *i*-tes Wort, so wird **`null`** zurückgeliefert.

a) Definieren Sie die oben beschriebene `FalschesFormatException`.
b) Definieren Sie die oben beschriebene statische Methode `berechneWert()`.

Hinweis: Die statische Methode **`int`** `parseInt(String s)` der Klasse `Integer` liefert den **`int`**-Wert der als String repräsentierten Zahl `s` zurück. Falls der übergebene String `s` keinen **`int`**-Wert repräsentiert, wird eine `NumberFormatException` geworfen.

Aufgabe 44 (H) Ausgabe einer Textdatei (2 Punkte)

Implementieren Sie ein Java-Programm `Cat.java`, das Dateien auf dem Bildschirm anzeigt. Es soll dazu den ersten Kommandozeilenparameter als Dateinamen interpretieren. Diese Datei soll nun zum lesen geöffnet werden, und Zeile für Zeile am Bildschirm ausgegeben werden. Sehen Sie Sich dazu in der Java API Dokumentation (`http://java.sun.com/j2se/1.5.0/docs/api/`) die Klassen `java.io.FileReader` und `java.io.BufferedReader` an.

Aufgabe 45 (H) Arbeiten mit Textdateien (12 Punkte)

Ziel dieser Aufgabe ist, ein Java Programm `TextStatistics.java` zu implementieren, welches Statistiken über Textdateien ausgibt. Der erste Kommandozeilenparameter soll den Namen der einzulesenden Datei, der zweite Parameter den Namen der Datei in welche die Ausgabe erfolgen soll, spezifizieren. Folgende Informationen sollen sowohl auf dem Bildschirm ausgegeben, als auch in eine Datei geschrieben werden:

a) Die Anzahl der Zeilen bis zum Ende der Datei.
b) Die Anzahl der Zeichen (ohne Leerzeichen).
c) Die Anzahl der verschiedenen Buchstaben (ohne Unterscheidung von Groß- und Kleinschreibung).
d) Die Anzahl der Wörter, wobei ein Wort als eine Zeichenkette, die nur aus Buchstaben und Zahlen besteht, aufzufassen ist.
e) Die Anzahl der Sätze. Ein Satz sei dabei eine Folge von Wörtern, die entweder mit einem Punkt, einem Fragezeichen oder einem Ausrufezeichen endet.
f) Die Summe aller vorhandenen Ziffern.

Hinweis: Lesen Sie vor dem Bearbeiten der Aufgaben die Dokumentation der Klassen `java.lang.Character` und `java.lang.String`. (`http://java.sun.com/j2se/1.5.0/docs/api/`)