Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Institut für Angewandte und Numerische Mathematik

Programmieren: Einstieg in die Informatik mit Java

WS 2006/2007

Dr. G. Bohlender
Dipl.–Math. techn. M. Richter

27.11.2006

Aufgabenblatt 5

Bearbeitungszeitraum: 30.11.2006 – 13.12.2006

Aufgabe 12 (Pflichtaufgabe): *Numerische Integration II*

In dieser Aufgabe wird das Programm `Quadratur` von Aufgabe 11 erweitert.

(a) Erweitern Sie die Methode `evalFunction` derart, dass zusätzlich die Funktionswerte der folgenden Funktionen zurück gegeben werden können:

$$f_4(x) = \frac{1}{1+x^2}, \qquad f_5(x) = \frac{e^{-\frac{1}{2}x^2}}{\sqrt{2\pi}}.$$

Den Wert der Konstanten π erhalten Sie unter Java mit dem Befehl `Math.PI`.

(b) Erstellen Sie eine statische Methode mit dem Namen `transform`, die eine Stelle z im sogenannten Referenzintervall $[-1, 1]$ auf eine Stelle x in einem beliebigen Intervall $[a, b]$ gemäß

$$x = \frac{1-z}{2}a + \frac{1+z}{2}b \tag{5}$$

abbildet. Übergeben Sie z, a und b als `double`-Argumente an die Methode. Die berechnete Stelle x soll als `double`-Wert zurück gegeben werden.

(c) Bislang können nur Integrale über das Referenzintervall $[-1, 1]$ numerisch berechnet werden. In diesem Aufgabenteil wird das Programm deshalb dahingehend erweitert, Integrale über beliebige Intervalle zu berechnen. Ändern Sie zu diesem Zweck die Methode `evalQuadratureRule` wie folgt ab:

Übergeben Sie zunächst die Randpunkte a und b eines Intervalls $[a, b]$ als zusätzliche `double`-Argumente an die Methode, und verwenden Sie die folgende Quadraturformel:

$$\frac{b-a}{2}\sum_{j=1}^{q} w_j f_k(\xi_j) \tag{6}$$

Die Gewichte $w_1, \ldots, w_q$ entnehmen Sie der der Tabelle aus Aufgabe 11. Die Stützstellen $\xi_1, \ldots, \xi_q$ erhalten Sie, indem Sie die Referenzstützstellen $\zeta_i, \ldots, \zeta_q$ aus Aufgabe 11 mit der Methode `transform` auf das Intervall $[a, b]$ abbilden.

Lesen Sie in der `main`-Methode zusätzlich die Randpunkte a und b von der Konsole ein berechnen Sie numerisch die Integrale der Funktionen f_1 bis f_5 über verschiedene Intervalle.

Hinweis für Interessierte: Die mathematische Grundlage für das Vorgehen in Aufgabenteil (c) ist die Substitutionsregel für Integrale. Mit der Substitution (5) gilt nämlich

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\int_{-1}^{1} f(x(z))\,dz.$$

Aufgabe 13: *Numerische Integration III*

Um Integrale numerisch noch besser anzunähern, unterteilt man ein Intervall $[a,b]$ üblicherweise in n kleinere Intervalle $[x_0,x_1],[x_1,x_2],\ldots,[x_{n-1},x_n]$, mit $x_0=a$ und $x_n=b$, und berechnet das Integral über jedes dieser Teilintervalle. Den Wert des Integrals über das gesamte Intervall $[a,b]$ erhält man, indem man alle berechneten Integralwerte über die Teilintervalle aufaddiert.
Setzen Sie diese Idee in Ihrem Programm `Quadratur` um. Schreiben Sie dazu eine statische Methode mit dem Namen `evalIteratedQuadratureRule`, welche zu einer gegebenen Zahl n, einer Ordnung q, einem Funktionsindex k, sowie zwei Randpunkten a und b die Integrale

$$\int_{x_{i-1}}^{x_i} f_k(x)\,dx \tag{7}$$

für $i=1,\ldots,n$ durch eine Quadraturformel der Ordnung q annähert und die genäherten Werte aufsummiert. Wählen Sie dabei

$$x_i = a + i\,\frac{b-a}{n}, \tag{8}$$

und verwenden Sie die Methode `evalQuadratureRule`! Das Ergebnis soll als `double`-Wert zurück gegeben werden.
Lesen Sie in der `main`-Methode zusätzlich die Anzahl der Teilintervalle n von der Konsole ein und geben Sie das Ergebnis von `evalIteratedQuadratureRule` auf den Bildschirm aus.
Untersuchen Sie anhand Ihres fertigen Programms, wie sich die Anzahl der Teilintervalle n und die Ordnung q der verwendendeten Quadraturformeln auf die Genauigkeit der berechneten Integrale auswirkt. Sie können dazu beispielsweise die folgenden zwei Integrale verwenden:

$$\int_{-1}^{1} f_4(x)\,dx \approx 1.570796326794897 \qquad\qquad \int_{-1}^{1} f_5(x)\,dx \approx 0.682689492137086$$

Aufgabe 14 (Pflichtaufgabe): *Eingabe aus Dateien / Statistik*

In den bisherigen Aufgaben wurden Eingabedaten für Java-Programme stets von der Konsole eingelesen. Diese Vorgehensweise ist jedoch nicht praktikabel, falls ein Programm umfangreiche Eingabedaten benötigt. Deshalb erfahren Sie in dieser Aufgabe, wie man Eingabedaten aus Dateien einliest.

(a) Erstellen Sie ein Java-Programm mit dem Namen `Statistik`. Importieren Sie in dieses Programm die Klassenbibliotheken `java.util.*` und `java.io.*`. Dies ist notwendig um

Daten aus Dateien einlesen zu können. Erweitern Sie außerdem die Definition der `main`-Methode um die Anweisung `throws FileNotFoundException`. Ihr Programm sollte also wie folgt aussehen:

```
import java.util.*;
import java.io.*;

public class Statistik {
  public static void main(String[] args) throws FileNotFoundException {
    Locale.setDefault(Locale.US);
    Scanner sc;
    /* Anweisungen */
  }
}
```

(b) Laden Sie von der Homepage der Vorlesung

`www.mathematik.uni-karlsruhe.de/ianm2/lehre/java2006w`

die Textdatei `xetra-okt.txt` herunter, und speichern Sie diese in demselben Ordner ab, in dem sich auch Ihr Java-Programm befindet. In dieser Datei sind die Tagesschlusskurse des *Xetra*®-Handelssystems der *Deutschen Börse Frankfurt* für den Monat Oktober 2006 als `double`-Werte gespeichert. Öffnen Sie in Ihrem Programm diese Datei mit einem Scanner mit dem Befehl

```
sc = new Scanner(new File("xetra-okt.txt"));
```

Sie können den Scanner nun wie gewohnt verwenden. Die Daten werden jetzt jedoch nicht von der Konsole, sondern aus der Datei `xetra-okt.txt` gelesen. Mit dem Befehl `sc.nextDouble()` können Sie also nacheinander die einzelnen Tagesschlusskurse in Ihr Programm einlesen. Die Methode `sc.hasNextDouble()` gibt den Wahrheitswert `true` zurück, wenn noch ein weiterer Kurs gelesen werden kann. Mit `sc.close()` schließen Sie die Datei wieder.

(c) Bestimmen Sie die *Anzahl* der Tagesschlusskurse n, deren *Mittelwert* $\overline{x}$ sowie deren *Streuung* s. Berechnen Sie zuerst n, danach $\overline{x}$ und danach s. Öffnen Sie für jede Berechnung die Datei `xetra-okt.txt` von Neuem und lesen Sie die einzelnen Tagesschlusskurse $x_1, \ldots, x_n$ der Reihe nach ein. Verwenden Sie dazu eine `while`-Schleife zusammen mit der Methode `sc.hasNextDouble()`. Für den Mittelwert und die Streuung gelten folgende Formeln:

$$\overline{x} = \sum_{i=1}^{n} \frac{x_i}{n}$$

$$s = \sqrt{\sum_{i=1}^{n} \frac{(x_i - \overline{x})^2}{n-1}}$$

Hinweis: Sie sollten folgende Werte als Ergebnis herausbekommen: $n = 22$, $\overline{x} \approx 6161.30$, $s \approx 89.22$