Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Einführung in die Informatik
Sommersemester 2015

Prof. Dr. Wolfram Burgard
Jörg Röwekämper
Alexander Schiotka

Übungsblatt 10

Abgabe bis Freitag, 10.07.2015, 10:00 Uhr

Hinweis:

Aufgaben immer per E-Mail (eine E-Mail pro Blatt und Gruppe) an den zuständigen Tutor schicken (Bei Programmieraufgaben Java Quellcode und evtl. benötigte Datendateien).

Aufgabe 10.1

Zwei Matrizen $A, B \in \mathbb{R}^{n \times n}$ werden wie folgt multipliziert[1]:

$$C = AB$$
$$C_{ij} = \sum_{k=0}^{i-1} A_{ik} B_{kj}$$

Hinweis: Auf der Vorlesungswebseite finden Sie eine vorhandene Klasse `Matrix`. Nutzen Sie diese für die Lösung der folgenden Aufgaben.

1. Implementieren Sie eine Methode `public Matrix mult(Matrix b)` zur Multiplikation zweier Matrizen, die das Ergebnis als Objekt der Klasse `Matrix` zurückgibt.
2. Nutzen Sie das Projekt `MatrixProject`[2] von der Vorlesungswebseite, um Ihre implementierte Methode aus der ersten Teilaufgabe mit JUnit zu testen.
3. Führen Sie eine Aufwandsabschätzung der implementierten Methode aus Teilaufgabe eins in Abhängigkeit von der Größe n der Matrix durch.

[1] Matrix-Multiplikation: `https://en.wikipedia.org/wiki/Matrix_multiplication#Matrix_product_.28two_matrices.29`

[2] MatrixProject: `http://ais.informatik.uni-freiburg.de/teaching/ss15/info/exercices/`

Aufgabe 10.2

Jede symmetrische positiv definite Matrix $A \in \mathbb{R}^{n \times n}$ kann eindeutig in der Form

$$A = LL^T$$

geschrieben werden. Dabei ist L eine untere Dreiecksmatrix und kann mittels Cholesky-Zerlegung bestimmt werden [3]. Die Elemente der Dreiecksmatrix L lassen sich wie folgt bestimmen:

$$\begin{aligned} L_{ii} &= \sqrt{a_{ii} - \sum_{k=0}^{i-1} L_{ik}^2} \\ L_{ji} &= \frac{1}{L_{ii}} \left(a_{ij} - \sum_{k=0}^{i-1} L_{ik} L_{jk} \right) \qquad j = i+1, i+2, \ldots, n-1 \end{aligned}$$

Implementieren Sie diese Zerlegung. Erweitern Sie dazu die Klasse `Matrix` (siehe Homepage) um die Methode `public Matrix chol()`. Führen Sie anschließend eine Aufwandsabschätzung Ihrer Implementierung in Abhängigkeit von n durch.

Aufgabe 10.3

Betrachten Sie die folgende Funktion $f : \mathbb{N}_0 \to \mathbb{N}_0$ ($\mathbb{N}_0$ sind die natürlichen Zahlen inklusive 0):

$$f(n) = \begin{cases} 0 & , \ n = 0 \\ 1 & , \ n = 1 \\ f(n-1) + f(n-2) & , \ n > 1 \end{cases}$$

1. Schreiben Sie eine rekursive Java-Methode, die die Funktion f implementiert.
2. Schreiben Sie eine *nicht*-rekursive Java-Methode, die die Funktion f implementiert.

Aufgabe 10.4

Betrachten Sie den folgenden Algorithmus:

```
static int f(int x, int y) {

  if (y > x) {
    return f(y, x);   // Zeile 4
  }
  int z = x % y;

  if (z == 0) {
    return y;         // Zeile 9
  } else {
    return f(y, z);   // Zeile 11
  }
}
```

Zeichnen Sie die Activation Records für den Aufruf $f(12, 21)$ zu dem Zeitpunkt, an dem die maximale Rekursionstiefe erreicht ist.

[3] siehe auch „Taschenbuch der Mathematik “, Bronstein, Semendjajew, Musiol und Mühlig