
Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (2 Punkte):
Aufgabe 2 (2 Punkte):
Aufgabe 3 (3 Punkte):
Aufgabe 4 (2 Punkte):
Aufgabe 5 (1 Punkte):
Aufgabe 6 (2 Punkte):
Aufgabe 7 (5 Punkte):
Aufgabe 8 (3 Punkte):
Aufgabe 9 (4 Punkte):
Aufgabe 10 (6 Punkte):

Gesamtpunktzahl:

Schriftlicher Test zu C (90 Minuten)
VU Einführung ins Programmieren für TM

06. Mai 2011

Aufgabe 1 (2 Punkte). Schreiben Sie einen Struktur-Datentyp `LMatrix` zur Speicherung von unteren Dreiecksmatrizen $A \in \mathbb{R}^{n \times n}$ beliebiger Dimension $n \in \mathbb{N}$, d.h. $A_{jk} = 0$ für $j < k$. In der Struktur sollen neben der Dimension n die Koeffizienten A_{jk} in Form eines geeigneten `double`-Vektor gespeichert werden. Die offensichtlichen Nulleinträge der Matrix sollen *nicht* gespeichert werden, d.h. der Vektor hat die Länge $\frac{n(n+1)}{2} = \sum_{j=1}^n j$.

ACHTUNG: Diese Struktur soll auch in allen nachfolgenden Aufgaben verwendet werden.

Aufgabe 2 (2 Punkte). Erklären Sie, an welcher Stelle a_ℓ des in der Struktur gespeicherten Vektors $a \in \mathbb{R}^N$ mit $N = \frac{n(n+1)}{2} = \sum_{j=1}^n j$ ein Matrix-Eintrag A_{jk} gespeichert wird.

Aufgabe 3 (3 Punkte). Schreiben Sie eine Funktion `newLMatrix`, die für gegebenes $n \in \mathbb{N}$ eine untere Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ allokiert und initialisiert.

Aufgabe 4 (2 Punkte). Schreiben Sie eine Funktion `delMatrix`, die den Speicher einer mittels `newMatrix` angelegten unteren Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ freigibt und den `NULL`-Pointer zurückgibt.

Aufgabe 5 (1 Punkt). Schreiben Sie eine Funktion `getMatrixDimension`, die die Dimension $n \in \mathbb{N}$ einer unteren Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ zurückgibt.

Aufgabe 6 (2 Punkte). Schreiben Sie eine Funktion `getLMatrixCoefficient`, die für gegebene Indizes j, k den Koeffizienten A_{jk} einer unteren Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ zurückgibt. Berücksichtigen Sie explizit den Fall $j < k$, für den nach Definition $A_{jk} = 0$ gilt.

Aufgabe 7 (5 Punkte). Schreiben Sie eine Funktion `computeLMatrixNorm`, die die Norm

$$\|A\| := \max_{j=0,\dots,n-1} \sum_{k=0}^{n-1} |A_{jk}|$$

einer unteren Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ berechnet und zurückgibt. Die offensichtlichen Nulleinträge der Matrix $A_{jk} = 0$ für $j < k$ sollen dabei nicht ausgewertet und addiert werden, um den Aufwand so gering wie möglich zu halten.

Aufgabe 8 (3 Punkte). Was versteht man unter „Aufwand“ eines Algorithmus? Welchen Aufwand hat Ihre Lösung zu Aufgabe 7? Begründen Sie Ihre Antwort und geben Sie den Aufwand zusätzlich in Landau-Notation $\mathcal{O}(\cdot)$ an. Was bedeutet das für die Laufzeit Ihrer Implementierung in Abhängigkeit der Dimension $n \in \mathbb{N}$?

Aufgabe 9 (4 Punkte). Das Produkt $C = AB \in \mathbb{R}^{n \times n}$ zweier unterer Dreiecksmatrizen $A, B \in \mathbb{R}^{n \times n}$ ist wieder eine untere Dreiecksmatrix. Beweisen Sie diese Aussage mathematisch, indem Sie die Laufindizes des Matrixproduktes

$$C_{ik} = \sum_{j=0}^{n-1} A_{ij}B_{jk} \quad \text{für } i, k = 0, \dots, n-1$$

kritisch betrachten.

Aufgabe 10 (6 Punkte). Schreiben Sie eine Funktion `computeLMM`, die die Produktmatrix $C = AB \in \mathbb{R}^{n \times n}$ zweier unterer Dreiecksmatrizen $A, B \in \mathbb{R}^{n \times n}$ berechnet und in der `LMatrix`-Struktur ablegt. Dazu dürfen Sie die Funktion

```
void setLMatrixEntry(LMatrix* A, int j, int k, double Ajk)
```

verwenden. Auf die Nulleinträge von A und B soll aus Effizienzgründen nicht zugegriffen werden.

