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

03. Oktober 2011

Aufgabe 1 (2 Punkte). Schreiben Sie einen Struktur-Datentyp `UMatrix` zur Speicherung von oberen Dreiecksmatrizen $A \in \mathbb{R}^{n \times n}$

$$A = \begin{pmatrix} A_{00} & A_{01} & A_{02} & \cdots & A_{0,n-1} \\ 0 & A_{11} & A_{12} & \cdots & A_{1,n-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & A_{n-2,n-1} \\ 0 & \cdots & \cdots & 0 & A_{n-1,n-1} \end{pmatrix}$$

beliebiger Zeilenanzahl $n \in \mathbb{N}$, d.h. $A_{jk} = 0$ für $j > k$. In der Struktur sollen neben der Zeilenanzahl n die Koeffizienten A_{jk} in Form eines geeigneten `double`-Vektors $a \in \mathbb{R}^N$ gespeichert werden. Die offensichtlichen Nulleinträge der Matrix sollen *nicht* gespeichert werden, d.h. der Vektor a hat die Länge $N = \sum_{j=1}^{n} j = \frac{n(n+1)}{2}$.

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. Geben Sie eine Formel an, die den Index ℓ in Abhängigkeit von j und k angibt.

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

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

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

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

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

$$\|A\| := \max_{k=0,\dots,n-1} \Big(\sum_{j=0}^{n-1} |A_{jk}|^2 \Big)^{1/2}$$

einer oberen 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 Zeilenanzahl $n \in \mathbb{N}$?

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

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

kritisch betrachten und geeignet vereinfachen.

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

```
void setUMatrixEntry(UMatrix* A, int j, int k, double Ajk)
```

verwenden. Achten Sie explizit darauf, auf die Nulleinträge von A und B nicht zuzugreifen.