Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
25.-27.11.2015

**Übungen zur Vorlesung**
**Einführung in das Programmieren für TM**

**Serie 8**

**Aufgabe 8.1.** Die Frobeniusnorm einer Matrix $A \in \mathbb{R}^{m\times n}$ ist durch

$$\|A\|_F := \Big(\sum_{j=1}^{m}\sum_{k=1}^{n} A_{jk}^2\Big)^{1/2}$$

definiert. Schreiben Sie eine Funktion `frobeniusnorm`, die für gegebene Matrix $A$ und gegebene Dimensionen $m, n \in \mathbb{N}$ die Frobeniusnorm berechnet. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem die Zeilen- und Spaltendimensionen $m, n \in \mathbb{N}$ und $A$ eingelesen werden und $\|A\|_F$ ausgegeben wird. Die Matrix $A$ soll dabei als dynamische Matrix (vom Typ `double**`) realisiert werden. Speichern Sie den Source-Code unter `frobeniusnorm.c` in das Verzeichnis `serie08`.

**Aufgabe 8.2.** Schreiben Sie eine Funktion `void unique(double * x, int * n)`, die einen Vektor $x \in \mathbb{R}^n$ aufsteigend sortiert, doppelte Einträge streicht und den Vektor in gekürzter Form zurückgibt. Die Funktion soll also beispielsweise den Vektor $x = (4, 3, 5, 1, 4, 3, 4) \in \mathbb{R}^7$ durch den Vektor $x = (1, 3, 4, 5) \in \mathbb{R}^4$ überschreiben. Die Länge $n$ der Vektoren ist dynamisch zu realisieren. Schreiben Sie ein aufrufendes Hauptprogramm, in dem $n$ und $x \in \mathbb{R}^n$ eingelesen werden und das Ergebnis der Funktion `unique` ausgegeben wird. Speichern Sie den Source-Code unter `unique.c` in das Verzeichnis `serie08`.

**Aufgabe 8.3.** Schreiben Sie eine Funktion `checkoccurrence`, die einen String $s$ und einen Buchstaben $b$ übernimmt und zurückgibt wie oft $b$ in $s$ vorkommt. Dabei zähle man sowohl das Vorkommen als Großbuchstabe als auch das Vorkommen als Kleinbuchstabe. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem $s$ und $b$ eingelesen und checkoccurrence aufgerufen werden. Speichern Sie den Source-Code unter `checkoccurrence.c` in das Verzeichnis `serie08`.

**Aufgabe 8.4.** Für eine konvergente Folge $(x_n)_{n\in\mathbb{N}}$ mit Grenzwert $x$ spricht man von Konvergenzordnung $p \geq 1$, falls es eine Konstante $c > 0$ gibt mit $|x_n - x| \leq c\,|x_{n-1} - x|^p$ für alle $n \in \mathbb{N}$. Mit dem Ansatz

$$|x_{n+2} - x| = c|x_{n+1} - x|^p \quad \text{und} \quad |x_{n+1} - x| = c|x_n - x|^p \quad \text{für } n \in \mathbb{N}$$

kann man für fixiertes $n$ die Unbekannten $p$ und $c$ bestimmen. Elementare Rechnung zeigt dann

$$p = \frac{\log(|x_{n+2} - x|/|x_{n+1} - x|)}{\log(|x_{n+1} - x|/|x_n - x|)} \quad \text{und} \quad c = \frac{|x_{n+2} - x|}{|x_{n+1} - x|^p},$$

d.h. aus den Gleichungen für die Fehler $|x_{n+2} - x|$ und $|x_{n+1} - x|$ berechnet man die Unbekannten $p_n := p$ und $c_n := c$. Leiten Sie diese Gleichheiten her! Schreiben Sie eine Funktion `convorder`, die für eine gegebene Folge $(x_n)_{n=1}^N$ und einen Grenzwert $x$ die experimentelle Konvergenzrate $p, c \in \mathbb{R}^{N-2}$ berechnet und zurückgibt. — Üblicherweise ist $x$ unbekannt und man hat nur eine Folge $(x_n)_{n=1}^N$ von Approximationen. In diesem Fall kann man die Funktion `convorder` mit der Teilfolge $(x_n)_{n=1}^{N-1}$ und $x := x_N$ verwenden. Testen Sie Ihre Implentierung indem Sie die experimentelle Konvergenzrate des Newton-Verfahrens (Serie 7, Aufgabe 7.5) berechnen. Speichern Sie den Source-Code unter `convorder.c` in das Verzeichnis `serie08`.

**Aufgabe 8.5.** Schreiben Sie eine Bibliothek zur Verwaltung von *spaltenweise* gespeicherten $m \times n$-Matrizen. Implementieren Sie die folgenden Funktionen

- `double* mallocmatrix(int m, int n)`
  Allokieren von Speicher für eine spaltenweise gespeicherte $m \times n$- Matrix.
- `double* freematrix(double* matrix)`
  Freigeben des allokierten Speichers einer Matrix.

- `double* reallocmatrix(double* matrix, int m, int n, int mNew, int nNew)`
  Reallokieren und initialisieren von neuen Einträgen.

Speichern Sie die Funktionssignaturen in das Header-File `dynamicmatrix.h`. Schreiben Sie auch entsprechende Kommentare zu den Funktionen in das Header-File. In die Datei `dynamicmatrix.c` kommt dann die Implementierung der Funktionen. Verwenden Sie dynamische Arrays.

**Aufgabe 8.6.** Erweitern Sie die Bibliothek aus Aufgabe 8.5 um folgende Funktionalitäten

- `void printmatrix(double* matrix, int m, int n)`
  Gibt eine spaltenweise gespeicherte $m \times n$-Matrix als Matrix am Bildschirm aus. Die $2 \times 3$-Matrix `double matrix[6]={1,2,3,4,5,6}` soll wie folgt ausgegeben werden:

```
1 3 5
2 4 6
```

- `double* scanmatrix(int m, int n)`
  Allokiert Speicher für eine Matrix und liest die Koeffizienten der Matrix von der Tastatur ein.
- `double* cutOffRowJ(double* matrix, int m, int n, int j)`
  Schneidet die $j$-te Zeile aus einer $m \times n$-Matrix heraus.
- `double* cutOffColK(double* matrix, int m, int n, int k)`
  Schneidet die $k$-te Spalte aus einer $m \times n$-Matrix heraus.

Verwenden Sie dynamische Arrays. Schreiben Sie ein main-Programm um die Funktionen aus dieser Aufgabe und aus Aufgabe 8.5 zu testen.

**Aufgabe 8.7.** Für eine Matrix $A \in \mathbb{R}^{m \times n}$ ist die Zeilensummennorm durch

$$\|A\| = \max_{j=1,\ldots,m} \sum_{k=1}^{n} |A_{jk}|$$

gegeben. Schreiben Sie eine Funktion `zeilensummennorm`, die die Zeilensummennorm einer Matrix $A$ berechnet, die spaltenweise gespeichert ist. Schreiben Sie ein aufrufendes Hauptprogramm, in dem $A$ eingelesen und $\|A\|$ ausgegeben wird. Verwenden Sie die Funktionen aus der Bibliothek aus Aufgabe 8.5 und Aufgabe 8.6. Speichern Sie den Source-Code unter `zeilensummennorm.c` in das Verzeichnis `serie08`.

**Aufgabe 8.8.** Was ist ein Gleitkommazahlsystem? Aus welchen Bestandteilen setzt sich eine Gleitkommazahl zusammen? Wie bestimmt man daraus ihren Wert? Was verbirgt sich hinter den Symbolen `Inf`, `-Inf` und `NaN`? Was ist die Maschinengenauigkeit `eps`? Was ist eine normalisierte Gleitkommazahl? Was ist ein implizites erstes Bit?