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. The Frobenius-norm of a matrix $A \in \mathbb{R}^{m\times n}$ is defined by

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

Write a function `frobeniusnorm` which computes the Frobenius norm of a given matrix A. Furthermore, write a main program that reads in the dimensions m, n and the matrix A. The matrix should be stored as a dynamic matrix (of type `double**`). Save your source code as `frobeniusnorm.c` into the directory `serie08`.

Aufgabe 8.2. Write a function `void unique(double * x, int * n)` which reads in a vector $x \in \mathbb{R}^n$, sorts this vector in ascending order, eliminates entries that appear more than once, and returns the shortened vector. For instance, the vector $x = (4, 3, 5, 1, 4, 3, 4) \in \mathbb{R}^7$ should be replaced by the vector $x = (1, 3, 4, 5) \in \mathbb{R}^4$. Write a main program that reads in the length $n \in \mathbb{N}$ and the vector $x \in \mathbb{R}^n$, and prints out the shortened vector. Work with dynamically allocated memory. Save your source code as `unique.c` into the directory `serie08`.

Aufgabe 8.3. Write a function `checkoccurrence`, which, given a string s and a character b, returns how many times b occurs in s. Both the lowercase and the uppercase versions of b contribute to the number of occurrences. Then, write a main program which reads s and b from the keyboard and calls the function. Save your source code as `checkoccurrence.c` into the directory `serie08`.

Aufgabe 8.4. Given a convergent sequence $(x_n)_{n\in\mathbb{N}}$ with limit x, if there exist $p \geq 1$ and a constant $c > 0$ such that $|x_n - x| \leq c\,|x_{n-1} - x|^p$ for all $n \in \mathbb{N}$, then we say that p is the convergence rate of $(x_n)_{n\in\mathbb{N}}$ towards x. With the ansatz

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

we can determine the values of p and c for any n. An easy computation shows

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

To start with, derive the above formulas. Then, write a function `convorder`, which, given a sequence $(x_n)_{n=1}^N$ and a limit x, computes and returns the empirical convergence rate $p, c \in \mathbb{R}^{N-2}$. In concrete situations, the limit x is usually unknown and only the sequence $(x_n)_{n=1}^N$ is available. In this case, the function `convorder` should apply the above formulas to the subsequence $(x_n)_{n=1}^{N-1}$ and $x := x_N$. Test your implementation by computing the empirical convergence rate of the Newton method (Serie 7, Exercise 7.5). Save your source code as `convorder.c` into the directory `serie08`.

Aufgabe 8.5. Write a library for *columnwise*(!) stored $m\times n$-matrices. Implement the following functions

- `double* mallocmatrix(int m, int n)`
  Allocates memory for a columnwise stored $m \times n$ matrix.
- `double* freematrix(double* matrix)`
  Frees memory of a matrix.
- `double* reallocmatrix(double* matrix, int m, int n, int mNew, int nNew)`
  Reallocates memory and initializes new entries.

Store the signatures of the functions in the header file `dynamicmatrix.h`. Write also appropriate comments to this functions in the header file. The file `dynamicmatrix.c` should contain the implementations of the above functions. Use dynamical arrays.

Aufgabe 8.6. Expand the library from Exercise 8.5 by the following functions.

- `void printmatrix(double* matrix, int m, int n)`
  Prints the column-wise-saved $m \times n$-Matrix on screen. The 2×3-Matrix `double matrix[6]={1,2,3,4,5,6}` shall look like in the following example:

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

- `double* scanmatrix(int m, int n)`
  Allocates memory for a matrix and scans the coefficients from keyboard-entry.
- `double* cutOffRowJ(double* matrix, int m, int n, int j)`
  Cuts off the j-th line from a $m \times n$-Matrix.
- `double* cutOffColK(double* matrix, int m, int n, int k)`
  Cuts off the k-th column from a $m \times n$-Matrix.

Use dynamical arrays. Write a main program, that tests the functions from this exercise and from Exercise 8.5.

Aufgabe 8.7. The *row-sum norm* of a matrix $A \in \mathbb{R}^{m\times n}$ is defined by

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

Write a function `rowsumnorm`, which computes the row-sum norm of a columnwise stored matrix A. Furthermore, write a main program that reads in A and computes $\|A\|$ thereof. Use the functions from the library from exercise 8.5 and exercise 8.6. Save your source code as `rowsumnorm.c` into the directory `serie08`..

Aufgabe 8.8. What is the system of floating-point numbers? Which parts does a floating-point number consist of? How can you determine its value? What is the meaning of `Inf`, `-Inf`, and `NaN`? What is the machine accuracy `eps`? What is a normalized floating-point number? What is a first implicit bit?