Valentin Helml
Dirk Praetorius
Julian Streitberger

Summer term 2022
1 April 2022

Exercise
Scientific programming in mathematics

Series 5

Exercise 5.1. Write a function `void bubbleSort(double* x, int n)`, which sorts a given vector $x \in \mathbb{R}^n$ in ascending order using the *bubble sort* algorithm, see Bubble sort. Use `assert` to ensure that $n \geq 1$. Moreover, write a main program that provides the input vector, calls the function, and prints both the input vector and the sorted vector to the screen. The length $n \in \mathbb{N}$ should be constant in the main program, but the function `bubbleSort` should be implemented for arbitrary n. Save your source code as `bubbleSort.c`. How did you test your code? Please formulate appropriate test cases to validate your code! What is the computational complexity of your implementation of `bubbleSort`? Justify your answer! What is the advantage of `bubbleSort` over `selectionSort` from the lecture?

Exercise 5.2. Write a function `geometricMean` that computes and returns the geometric mean value $\overline{x}_{\text{geom}}$ of a vector $x \in \mathbb{R}^n_{\geq 0}$ defined by

$$\overline{x}_{\text{geom}} := \sqrt[n]{\prod_{j=1}^{n} x_j}.$$

The length $n \in \mathbb{N}$ should be a constant in the main program, but the function `geometricMean` should be implemented for arbitrary n. Use `assert` to ensure that $n \geq 1$. Furthermore, write a main program that provides the input vector `x`, calls the function, and prints the geometric mean $\overline{x}_{\text{geom}}$ to the screen. Save your source code as `geometricMean.c`. How did you test your code? Please formulate appropriate test cases to validate your code!

Exercise 5.3. Write a function `double cubeRoot(double x, double precision)` which computes and returns the cubic root y of a given $x \in \mathbb{R}$ with a given precision, i.e., it holds that $|y^3 - x| \leq$ precision. Use suitable loops. You must not use `pow` or `cbrt`. To test your code, write a main program `cubeRoot.c` that compares the result of `cubeRoot` with the function `cbrt` from the mathematical library. How did you test your code? Please formulate appropriate test cases to validate your code!

Exercise 5.4. Write a function `lcm(int a, int b)`, which computes and returns the least common multiple of two given natural numbers $a, b \in \mathbb{N}$. Use `assert` to ensure that $a, b \geq 1$. Moreover, write a main program `lcm.c`, which reads $a, b \in \mathbb{N}$ from the keyboard and prints their least common multiple to the screen. How did you test your code? Please formulate appropriate test cases to validate your code!

Exercise 5.5. One way (not the best way) to approximate the number π is based on the so-called *Leibniz formula*

$$\pi = \sum_{k=0}^{\infty} \frac{4(-1)^k}{2k+1}.$$

In particular, for any $n \in \mathbb{N}$, the n-th partial sum S_n is defined by

$$S_n = \sum_{k=0}^{n} \frac{4(-1)^k}{2k+1}.$$

Then, S_n can be understood as an approximation of π, since $\lim_{n\to\infty} S_n = \pi$. Write a function `double partialSum(int n)` that computes S_n for a given number $n \in \mathbb{N}_0$. Use `assert` to ensure that $n \geq 0$. Moreover, write a main program `partialSum.c`, which reads $n \in \mathbb{N}_0$ from the keyboard and prints the resulting approximation S_n of π to the screen. How did you test your code? Please formulate appropriate test cases to validate your code!

Exercise 5.6. Write a main program `pascal.c`, which reads $n \in \mathbb{N}$ from the keyboard and appropriately prints the first n lines of Pascal's triangle to the screen by the following procedure: Every line starts and ends with a 1. The remaining entries are the sum of the two neighboring entries from the line above, e.g., for $n = 5$ the result should be

$$\begin{array}{ccccc} 1 & & & & \\ 1 & 1 & & & \\ 1 & 2 & 1 & & \\ 1 & 3 & 3 & 1 & \\ 1 & 4 & 6 & 4 & 1 \end{array}$$

For more details, see, e.g., Wikipedia. Use `assert` to ensure that $n \geq 1$. Save your source code as `pascal.c`.

Exercise 5.7. Write a function `void minmaxmean(double x[], int n)`, which computes and prints to the screen the minimum, the maximum, and the mean value $(1/n)\sum_{j=1}^{n} x_j$ of a given vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$. Additionally, write a main program `minmaxmean.c` that reads the vector $x \in \mathbb{R}^n$ from the keyboard and calls the function. The length of the vector should be constant in the main program, but the function `minmaxmean` should be programmed to work for arbitrary vector lengths. Use `assert` to ensure that `n` ≥ 1. How did you test your code? Please formulate appropriate test cases to validate your code!

Exercise 5.8. Write a function `char structure(double A[], int n)`, which takes a matrix $A \in \mathbb{R}^{n\times n}$ that is stored in column-wise order. The function should return

- `'d'` if A is diagonal (i.e. $A_{jk} = 0$ for all $j \neq k$),
- `'l'` if A is lower triangular (i.e. $A_{jk} = 0$ for all $j < k$),
- `'u'` if A is upper triangular (i.e. $A_{jk} = 0$ for all $j > k$).

Otherwise, the function should return `'f'` to indicate a full matrix. Determine the computational cost of your function.