Dirk Praetorius
Michele Ruggeri

Summer semester 2019
March 29, 2019

Scientific programming in mathematics

Exercise sheet 4

Arrays, `for` and `while` loops, and computational complexity

Exercise 4.1. Write a function `void cross(int n)`, which, given $n \in \{2, 3, \dots, 9\}$, prints a 'X' consisting of $2n - 1$ lines to the screen. For example, for $n = 6$, one should get the following output:

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

Use `assert` to ensure that $n \in \{2, 3, \dots, 9\}$. Write a main program, which reads in the parameter n from the keyboard and calls the function `cross`. How did you test the correctness of your code? Save your source code as `cross.c` into the directory `series04`.

Exercise 4.2. Write a function `geometricMean` that computes and returns the geometric mean value

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

of a given vector $x \in \mathbb{R}^n_{\geq 0}$. The length $n \in \mathbb{N}$ should be a constant in the main program, but the function `geometricMean` should be implemented for arbitrary lengths n. Furthermore, write a main program that provides the input vector, call the function, and prints to the screen its geometric mean. How did you test the correctness of your code? Save your source code as `geometricmean.c` into the directory `series04`.

Exercise 4.3. 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, given a matrix A and its dimensions $m, n \in \mathbb{N}$, computes and returns the Frobenius norm of the matrix. Furthermore, write a main program that provides the input parameters (the matrix A and its dimensions m, n), call the function, and prints to the screen the corresponding Frobenius norm $\|A\|_F$. The matrix should be stored columnwise as a vector of length mn. The dimensions $m, n \in \mathbb{N}$ should be constant in the main program, but the function `frobeniusNorm` should be programmed for arbitrary dimensions. What is the computational complexity of your implementation (for $n = m$)? Justify adequately your answer. How did you test the correctness of your code? Save your source code as `frobeniusnorm.c` into the directory `series04`.

Exercise 4.4. Write a function `maxCompare` that counts, given two vectors $a, b \in \mathbb{R}^n$, how often the global maximum of the vectors a and b denoted by $M := \max\{a_i, b_i \mid i = 1, \dots, n\}$ is represented in a and b at the same position. Accordingly, if M appears only in a or b, then the function should return 0. For example, for the vectors $a = (1.1, 4, 0.27, 4, 4, 3, 4, -1.5)$ and $b = (2.2, 4, 4, 0.0002, 4, -1, 2.7, 4)$ in $\mathbb{R}^8$, we have $M = 4$ and the function returns 2, since $a_2 = b_2 = a_5 = b_5 = M = 4$. The length $n \in \mathbb{N}$ should be a constant in the main program, but the function `maxCompare` should be programmed for arbitrary lengths n. Furthermore, write a main program which provides the input parameters to `maxCompare` and calls the function. How did you test the correctness of your code? Save your source code as `maxcompare.c` into the directory `series04`.

Exercise 4.5. Write a function `double cubeRoot(double x, double precision)` which computes (and returns) the cubic root of a given $x \in \mathbb{R}$ with a given precision. Use suitable loops. To test the correctness of your code, write a main program that compares the results of `cubeRoot` with those obtained with the function `cbrt` from the mathematical library. In particular, `cbrt` is the only function of the mathematical library that you are allowed to use in your code. Save your source code as `cubic.c` into the directory `series04`.

Exercise 4.6. Write a function `int 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 $a, b \in \mathbb{N}$. Moreover, write a main program which reads $a, b \in \mathbb{N}$ from the keyboard and prints their least common multiple to the screen. How did you test the correctness of your code? Save your source code as `lcm.c` into the directory `series04`.

Exercise 4.7. Write a function `scanfPositive`, which asks the user for a positive number $\tau > 0$ and then returns it. The input request is repeated until the provided input $\tau \in \mathbb{R}$ is strictly positive, i.e., if the given value satisfies $\tau \leq 0$, then the input is ignored and the request is repeated. Additionally, write a main program which calls `scanPpositive`. Save your source code as `scanfpositive.c` into the directory `series04`.

Exercise 4.8. Write a function `int sqrtBoundaries(double x)`, which, given $x \in \mathbb{R}_{\geq 0}$, computes and returns the unique $k \in \mathbb{N}_0$ satisfying $k \leq \sqrt{x} < k + 1$. You are not allowed to use the mathematical library in your code. In particular, you must use neither `sqrt` nor rounding functions (e.g., `floor` or `ceil`). Use `assert` to ensure that the input is admissible. Moreover, write a main program, which reads $x \in \mathbb{R}$ from the keyboard, calls `sqrtBoundaries` and print $k \in \mathbb{N}_0$ to the screen. Save your source code as `sqrtboundaries.c` into the directory `series04`.