Dirk Praetorius,
Gregor Gantner

Sommersemester 2018
17.04.2018

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 5

Aufgabe 5.1. Write a function `scanfpositive`, which asks the user for a positive number $\tau > 0$ and then returns it. The input request is repeated as long as $\tau \in \mathbb{R}$ is strictly positive, i.e., if the given value satisfies $\tau \leq 0$, then the input request is repeated. Additionally, write a main program, which calls `scanfpositive`. How did you test the correctness of your code? Save your source code as `scanfpositive.c` into the directory `serie05`.

Aufgabe 5.2. Write a *non-recursive* function `power`, which, given two real numbers $x > 1$ and $C > 0$, determines the smallest integer $n \in \mathbb{N}$ such that $x^n > C$. In your implementation, you are not allowed to use the function `log`. Use `assert` to ensure the conditions $x > 1$ and $C > 0$. Write a main program, which reads in x and C and prints out the corresponding n. Save your source code as `power` into the directory `serie05`.

Aufgabe 5.3. Write a function `lcm`, which computes for two given natural numbers $a, b \in \mathbb{N}$ the least common multiple and returns it. Additionally, write a main program, which reads in the numbers $a, b \in \mathbb{N}$ and prints the least common multiple on the screen. How did you test the correctness of your code? Save your source code as `lcm.c` into the directory `serie05`.

Aufgabe 5.4. The quotient sequence $(a_{n+1}/a_n)_{n\in\mathbb{N}}$ corresponding to the Fibonacci sequence $(a_n)_{n\in\mathbb{N}}$,

$$a_0 := 1, \quad a_1 := 1, \quad a_n := a_{n-1} + a_{n-2} \quad \text{für } n \geq 2,$$

converges towards the *golden ratio* $(1 + \sqrt{5})/2$. In particular, the difference sequence

$$b_n := \frac{a_{n+1}}{a_n} - \frac{a_n}{a_{n-1}}$$

converges towards 0. Write a *non-recursive* function `cauchy` that returns, for given $k \in \mathbb{N}$, the smallest $n \in \mathbb{N}$ such that $|b_n| \leq 1/k$. Moreover, write a main program that reads in $k \in \mathbb{N}$ and prints out the index $n \in \mathbb{N}$. How did you test the correctness of your code? Save your source code as `goldenRatio.c` into the directory `serie05`.

Aufgabe 5.5. For $x > 0$, the recursively defined sequence

$$x_1 := \frac{1}{2}(1 + x), \quad x_{n+1} := \frac{1}{2}\Big(x_n + \frac{x}{x_n}\Big) \quad \text{for } n \geq 1$$

converges towards $\sqrt{x}$. Write a *nonrecursive* function `sqrt_new` which computes for given $x > 0$ and $\tau > 0$ the *first* element x_n of the sequence that satisfies

$$\frac{|x_n - x_{n+1}|}{|x_n|} \leq \tau \quad \text{or} \quad |x_n| \leq \tau.$$

Check via `assert`, that $x > 0$. Moreover, write a main program which reads in x and τ, computes the approximation x_n of $\sqrt{x}$ and compares it to the exact value, i.e. prints out the absolute error $|x_n - \sqrt{x}|$. How did you test the correctness of your code? Save your source code as `sqrt_new.c` into the directory `serie05`.

Aufgabe 5.6. The cosine function can be represented as a series via

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!}.$$

The corresponding n-th partial sum is given by

$$C_n(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k}}{(2k)!}.$$

Write a *nonrecursive* function `cos_new`, which, given $x \in \mathbb{R}$ and $\tau > 0$, returns the first value of $C_n(x)$ such that

$$|C_n(x) - C_{n-1}(x)|/|C_n(x)| \leq \tau \quad \text{or} \quad |C_n(x)| \leq \tau.$$

Then, write a main program, which reads $x \in \mathbb{R}$ and $\tau > 0$ from the keyboard, calls the function and displays the computed value $C_n(x)$, as well as the value $\cos(x)$, the absolute error $|C_n(x) - \cos(x)|$ and the relative error $|C_n(x) - \cos(x)|/|\cos(x)|$ (provided $\cos(x) \neq 0$). Write the function in such a way that only one loop is needed and that x^{2k} and $(2k)!$ are computed in a cheap way. This means, you must not use specific functions to compute the power or the factorial. How did you test the correctness of your code? Save your source code as `cos.c` into the directory `serie05`.

Aufgabe 5.7. Consider the two series

$$a_N := \sum_{n=0}^{N} \frac{1}{(n+1)^2} \quad \text{und} \quad b_N := \sum_{n=0}^{N} \sum_{k=0}^{n} \frac{1}{(k+1)^2(n-k+1)^2}.$$

Write two functions that, given $N \in \mathbb{N}$, measure the time used for the computation of $(a_N)^2$ resp. b_N. What is the computational complexity (Aufwand) for the computation of $(a_N)^2$ resp. b_N? For instance: if the functions run 3 seconds for $N = 10^3$, how long does the computation take for in case of $N = 10^4$? Write a main program that, for different values of N, prints out the results tabularly. Do the results meet your expectations? How have you tested your implementation? Save your source code as `timing.c` into the directory `serie05`.

Aufgabe 5.8. Which types of comments do you know? What is the output of the following code and why?

```
#include <stdio.h>

/*int f(double x) {
    return (int) x;
  }
*/

main() {
  int x = 4;
  int y = 2*x*/* f(0.1)+3
              */1/4;
  // y = 1;
  printf("y = %d\n",y); // Print out result
}
```