Dirk Praetorius
Michele Ruggeri

Sommersemester 2015
05.05.2015

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 7

Aufgabe 7.1. Write a structure data-type `cdouble`, which stores the real part a and the imaginary part b of a complex number $a + bi \in \mathbb{C}$ as `double` variables. Implement the functions `cdouble* newCDouble(double a, double b)`, `cdouble* delCDouble(cdouble* c)` as well as `setCDoubleReal`, `getCDoubleReal`, `setCDoubleImag` sowie `getCDoubleImag`. Divide your source code into the header file `cdouble.h` and the file `cdouble.c`, and save it into the directory `serie07`.

Aufgabe 7.2. Write functions `cadd, csub, cmul, cdiv`, which realize addition, subtraction, multiplication, and division of complex numbers. Furthermore, implement the function `double cnorm(cdouble* c)`, which computes the squared norm $|a + ib|^2 := a^2 + b^2$. Use the structure from Exercise 7.1 to store complex numbers and use the corresponding `get` and `set` functions. Additionally, write a main program that reads two complex numbers $w, z \in \mathbb{C}$ from the keyboard and displays $|w|^2$, $|z|^2$, $w + z$, $w - z$, $w \cdot z$ as well as w/z (if $z \neq 0$). Write a main-progam to test the implemeted functions. Test your code on suitable examples. Save your source code as `carithmetik.c` into the directory `serie07`.

Aufgabe 7.3. Write a structure (data-type) `polynomial` for the storage of polynomials that are represented as $p(x) = \sum_{j=0}^{n} a_j x^j$. Note that you have to store the degree $n \in \mathbb{N}_0$ as well as the coefficient vector $(a_0, \dots, a_n) \in \mathbb{R}^{n+1}$. Write all necessary functions to work with this structure (`newPoly`, `delPoly`, `getPolyDegree`, `getPolyCoefficient`, `setPolyCoefficient`). Save your source code as `polynomial.c` into the directory `serie07`.

Aufgabe 7.4. The product $r = pq$ of two polynomials $p(x) = \sum_{j=0}^{m} a_j x^j$ and $q(x) = \sum_{j=0}^{n} b_j x^j$ is again a polynomial. Write a function `prodPoly` that computes the product r and stores it in the structure from Exercise 7.3. At first, think about the degree of the polynomial r. Additionally, write a main program that reads in two polynomials and computes the product thereof. Test your code on a suitable example. Save your source code as `prodPoly.c` into the directory `serie07`.

Aufgabe 7.5. Let $p(x) = \sum_{j=0}^{n} a_j x^j$ be a polynomial, given by the coefficient vector $a = (a_0, \dots, a_n) \in \mathbb{R}^{n+1}$. Write a function `evalpolynomial`, that, for a given coefficient vector a and a given evaluation point x, computes $p(x)$. `pow` must not be used to calculate x^j. Write a function, that only uses *one* loop. The degree of the polyomial denotes an fixed integer in the main program, but is an input parameter in the function function `evalpolynomial`.
Moreover, write a main-program, which reads the coefficients a_j and x from the keyboard, calls `evalpolynomial` and prints $p(x)$ on screen. Test your code on a suitable example. Save your source code as `evalPolynomial.c` into the directory `serie07`.

Aufgabe 7.6. The k-th derivative $p^{(k)}$ of a polynomial p is again a polynomial. Write a function `differentiatePolynomial` that computes the k-th derivative of a polynomial. For the storage of polynomials use the structure from Exercise 7.3. Additionally, write a main program that reads in p and k, and prints out $p^{(k)}$. Test your code on a suitable example. Save your source code as `differentiatePolynomial.c` into the directory `serie07`.

Aufgabe 7.7. Write a structure `CPoly` for the storage of polynomials, where the coefficients are complex numbers, i.e., $p(x) = \sum_{j=0}^{n} a_j x^j$ with $a_j \in \mathbb{C}$. The structure should contain the degree $n \in \mathbb{N}$ and the coefficients $(a_0, \dots, a_n) \in \mathbb{C}^{n+1}$. Use the structure from Exercise 7.1. Moreover, implement the functions `newCPoly`, `delCPoly`, `getCPolyDegree`, `getCPolyCoefficient`, and `setCPolyCoefficient`. Save your source code as `cpoly.c` into the directory `serie07`.

Aufgabe 7.8. Write a function `addCpolynomials` that computes the sum $r = p + q$ of two complex polynomials p, q and returns r. Use the structure from Exercise 7.7. Moreover, write a main program that reads in two polynomials p, q and prints out the sum $r = p + q$.
Test your code on a suitable example. Save your source code as `addcpoly.c` into the directory `serie07`.