Dirk Praetorius
Michele Ruggeri

Sommersemester 2015
05.05.2015

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 7

Aufgabe 7.1. Schreiben Sie einen Strukturdatentyp `cdouble`, in dem Realteil a und Imaginärteil b einer komplexen Zahl $a + bi \in \mathbb{C}$ jeweils als `double` gespeichert werden. Schreiben Sie Funktionen `cdouble* newCDouble(double a, double b)`, `delCDouble` sowie die vier Zugriffsfunktionen `setCDoubleReal`, `getCDoubleReal`, `setCDoubleImag` sowie `getCDoubleImag`. Speichern Sie den Source-Code, aufgeteilt in Header-Datei `cdouble.h` und `cdouble.c`, in das Verzeichnis `serie07`.

Aufgabe 7.2. Schreiben Sie Funktionen `cadd, csub, cmul, cdiv`, die die Addition, die Subtraktion, die Multiplikation und die Division für komplexe Zahlen realisieren. Weiters soll eine Funktion `double cnorm(cdouble* c)` programmiert werden, die das Betragsquadrat $|a + ib|^2 := a^2 + b^2$ zurückliefert. Verwenden Sie zur Speicherung die Struktur aus Aufgabe 7.1, und benutzen Sie beim Strukturzugriff nur die entsprechenden Zugriffsfunktionen. Schreiben Sie ein aufrufendes Hauptprogramm, in dem zwei komplexe Zahlen $w, z \in \mathbb{C}$ eingelesen werden und $|w|^2$, $|z|^2$, $w + z$, $w - z$, $w \cdot z$ sowie w/z (falls $z \neq 0$) ausgegeben werden. Testen Sie Ihren Code an geeigneten Beispielen. Speichern Sie den Source-Code unter `carithmetik.c` in das Verzeichnis `serie07`.

Aufgabe 7.3. Schreiben Sie einen Strukturdatentyp `polynomial` zur Speicherung von Polynomen, die bezüglich der Monombasis dargestellt sind, d.h. $p(x) = \sum_{j=0}^{n} a_j x^j$. Es ist also der Grad $n \in \mathbb{N}_0$ sowie der Koeffizientenvektor $(a_0, \dots, a_n) \in \mathbb{R}^{n+1}$ zu speichern. Schreiben Sie alle nötigen Funktionen, um mit dieser Struktur arbeiten zu können (`newPoly`, `delPoly`, `getPolyDegree`, `getPolyCoefficient`, `setPolyCoefficient`). Speichern Sie den Source-Code unter `polynomial.c` in das Verzeichnis `serie07`.

Aufgabe 7.4. Das Produkt $r = pq$ zweier Polynome $p(x) = \sum_{j=0}^{m} a_j x^j$ und $q(x) = \sum_{j=0}^{n} b_j x^j$ ist wieder ein Polynom. Schreiben Sie eine Funktion `prodPoly`, die das Produktpolynom r berechnet und in der Struktur aus Aufgabe 7.3 speichert. Überlegen Sie sich zunächst, welchen Grad das Polynom r hat und wie sich die Koeffizienten berechnen lassen. Schreiben Sie ein aufrufendes Hauptprogramm, in dem p und q eingelesen und $r = pq$ ausgegeben wird. Testen Sie Ihren Code an einem geeigneten Beispiel. Speichern Sie den Source-Code unter `prodPoly.c` in das Verzeichnis `serie07`.

Aufgabe 7.5. Gegeben sei ein Polynom $p(x) = \sum_{j=0}^{n} a_j x^j$ in Form seines Koeffizientenvektors $a = (a_0, \dots, a_n) \in \mathbb{R}^{n+1}$. Schreiben Sie eine Funktion `evalpolynomial`, die für gegebenen Koeffizientenvektor a und Auswertungspunkt x den Funktionswert $p(x)$ berechnet. Die Funktion `pow` zur Berechnung von x^j soll *nicht* verwendet werden. Schreiben Sie eine Funktion, die möglichst nur *eine* Schleife verwendet. Der Grad $n \in \mathbb{N}$ des Polynoms soll eine Konstante im Hauptprogramm sein, die Funktion `evalpolynomial` soll aber beliebigen Grad zulassen. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem die Koeffizienten a_j sowie der Auswertungspunkt x eingelesen werden und $p(x)$ ausgegeben wird. Testen Sie Ihren Code an einem geeigneten Beispiel. Speichern Sie den Source-Code unter `evalPolynomial.c` in das Verzeichnis `serie07`.

Aufgabe 7.6. Die k-te Ableitung $p^{(k)}$ eines Polynoms p ist wieder ein Polynom. Schreiben Sie eine Funktion `differentiatePolynomial`, die zu gegebenem p und $k \in \mathbb{N}$ die Ableitung $p^{(k)}$ berechnet. Zur Speicherung verwende man die Struktur aus Aufgabe 7.3. Schreiben Sie ein Hauptprogramm, das p und k einliest und $p^{(k)}$ ausgibt. Testen Sie Ihren Code an einem geeigneten Beispiel. Speichern Sie den Source-Code unter `differentiatePolynomial.c` in das Verzeichnis `serie07`.

Aufgabe 7.7. Schreiben Sie eine Struktur `CPoly` zur Speicherung von Polynomen mit komplexwertigen Koeffizienten, die bezüglich der Monombasis dargestellt sind, d.h. $p(x) = \sum_{j=0}^{n} a_j x^j$. Es sind also der Grad $n \in \mathbb{N}_0$ sowie der Koeffizientenvektor $(a_0, \dots, a_n) \in \mathbb{C}^{n+1}$ zu speichern. Verwenden Sie für die Darstellung der komplexwertigen Koeffizienten den Strukturdatentyp aus Aufgabe 7.1. Schreiben Sie die ferner die nötigen Zugriffsfunktionen `newCPoly`, `delCPoly`, `getCPolyDegree`, `getCPolyCoefficient` und `setCPolyCoefficient`. Speichern Sie den Source-Code unter `cpoly.c` in das Verzeichnis `serie07`.

Aufgabe 7.8. Schreiben Sie eine Funktion `addCpolynomials`, die die Summe $r = p + q$ zweier komplexer Polynome p und q (auch unterschiedlichen Grades) berechnet und zurückgibt. Verwenden Sie zur Speicherung die Struktur aus Aufgabe 7.7. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem zwei Polynome p, q eingelesen und die Summe $r = p + q$ ausgegeben werden. Testen Sie Ihren Code an einem geeigneten Beispiel. Speichern Sie den Source-Code unter `addcpoly.c` in das Verzeichnis `serie07`.