Dirk Praetorius,
Gregor Gantner

Sommersemester 2018
08.05.2018

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 8

Aufgabe 8.1. 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 `serie08`.

Aufgabe 8.2. Die Summe $r = p + q$ zweier Polynome p, q ist wieder ein Polynom. Schreiben Sie eine Funktion `addPolynomials`, die die Summe r berechnet. Zur Speicherung verwende man die Struktur aus Aufgabe 8.1. Zum Test schreibe man eine Funktion, die zwei Polynome einliest und deren Summe ausgibt. Speichern Sie den Source-Code unter `addPolynomials.c` in das Verzeichnis `serie08`.

Aufgabe 8.3. 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 8.1 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 `serie08`.

Aufgabe 8.4. Schreiben Sie eine Funktion `evalPoly`, die für ein gegebenes Polynom p und einen Punkt $x \in \mathbb{R}$ den Funktionswert $p(x)$ berechnet. Zur Speicherung verwende man die Struktur aus Aufgabe 8.1. Testen Sie Ihren Code an einem geeigneten Beispiel. Speichern Sie den Source-Code unter `evalPoly.c` in das Verzeichnis `serie08`.

Aufgabe 8.5. Die k-te Ableitung $p^{(k)}$ eines Polynoms p ist wieder ein Polynom. Schreiben Sie eine Funktion `differentiatePolynomial`, die zu gegebenem Polynom p und $k \in \mathbb{N}$ die Ableitung $p^{(k)}$ berechnet. Zur Speicherung von p und $p^{(k)}$ verwende man die Struktur aus Aufgabe 8.1. 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 `serie08`.

Aufgabe 8.6. Schreiben Sie einen Strukturdatentyp `cDouble`, in dem Realteil $a \in \mathbb{R}$ und Imaginärteil $b \in \mathbb{R}$ einer komplexen Zahl $z = a + bi \in \mathbb{C}$ jeweils als `double` gespeichert werden. Die imaginäre Einheit i erfüllt die Eigenschaft $i^2 = -1$, siehe

`https://de.wikipedia.org/wiki/Komplexe_Zahl`.

Schreiben Sie Funktionen

- `cDouble* newCDouble(double a, double b)`,
- `cDouble* delCDouble(cDouble* z)`

sowie die vier Zugriffsfunktionen

- `void setCDoubleReal(cDouble* z, double a)`,
- `double getCDoubleReal(cDouble* z)`,
- `void setCDoubleImag(cDouble* z, double b)`,
- sowie `double getCDoubleImag(cDouble* z)`.

Wie haben Sie Ihren Code auf Korrektheit getestet? Speichern Sie den Source-Code, aufgeteilt in Header-Datei `cdouble.h` und `cdouble.c`, in das Verzeichnis `serie08`.

Aufgabe 8.7. 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 $(a_0, \ldots, a_n) \in \mathbb{C}^{n+1}$ zu speichern. Verwenden Sie für die Darstellung der komplexwertigen Koeffizienten den Strukturdatentyp aus Aufgabe 8.6. Schreiben Sie ferner die nötigen Zugriffsfunktionen `newCPoly`, `delCPoly`, `getCPolyDegree`, `getCPolyCoefficient` und `setCPolyCoefficient`. Wie haben Sie Ihren Code auf Korrektheit getestet? Speichern Sie den Source-Code unter `cpoly.c` in das Verzeichnis `serie08`.

Aufgabe 8.8. 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 8.7. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem zwei Polynome p, q eingelesen und die Summe $r = p + q$ ausgegeben werden. Wie haben Sie Ihren Code auf Korrektheit getestet? Speichern Sie den Source-Code unter `addcpoly.c` in das Verzeichnis `serie08`.