

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. Schreiben Sie eine Funktion `evalPoly`, die für ein gegebenes Polynom p und Punkt $x \in \mathbb{R}$ den Funktionswert $p(x)$ ausrechnet. 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 8.3. 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 8.4. 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 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 8.5. 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 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 eine Funktion `evalDiffPoly`, die für ein gegebenes Polynom p , eine Ableitungsordnung $k \in \mathbb{N}_0$ und einen Punkt $x \in \mathbb{R}$ den Funktionswert $p^{(k)}(x)$ zurückgibt. Dabei soll das Polynom $p^{(k)}$ nicht explizit gebildet und gespeichert werden. Verwenden Sie für das Polynom p die Struktur aus Aufgabe 8.1. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem der Grad n , das Polynom p , die Ableitungsordnung k und der Punkt x eingelesen werden und $p^{(k)}(x)$ ausgegeben wird. Speichern Sie den Source-Code unter `evalDiffPoly.c` in das Verzeichnis `serie08`.

Aufgabe 8.7. 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 `serie08`.

Aufgabe 8.8. 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 8.7. Schreiben Sie die ferner die nötigen Zugriffsfunktionen `newCPoly`, `delCPoly`, `getCPolyDegree`, `getCPolyCoefficient` und `setCPolyCoefficient`. Speichern Sie den Source-Code unter `cpoly.c` in das Verzeichnis `serie08`.