Dirk Praetorius
Michele Ruggeri

Wintersemester 2014/15
09.12.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 11

Aufgabe 11.1. Schreiben Sie ein Programm, welches ein Wort einliest und überprüft ob es sich bei dem eingegeben Wort um ein Palindrom handelt. Ein Palindrom ist ein Wort, welches von vorne und hinten gelesen gleich lautet, z.B.: Anna, Otto, Reliefpfeiler. Speichern Sie den Source-Code unter `palindrom.c` in das Verzeichnis `serie11`.

Aufgabe 11.2. 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 `serie11`.

Aufgabe 11.3. 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 11.2, 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. Speichern Sie den Source-Code unter `carithmetik.c` in das Verzeichnis `serie11`.

Aufgabe 11.4. Schreiben Sie eine Struktur `CMatrix`, zur Speicherung von $(m \times n)$-Matrizen $A \in \mathbb{C}^{m\times n}$ mit komplexwertigen Koeffizienten. Benutzen Sie hierzu den Strukturdatentyp `cdouble` aus Aufgabe 11.2. Ferner schreibe man die zugehörigen Funktionen `newCMatrix`, `delCMatrix`, `getCMatrixM`, `getCMatrixN`, `getCMatrixCoeff`, `setCMatrixCoeff`.

Aufgabe 11.5. 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 11.2. Schreiben Sie die ferner die nötigen Zugriffsfunktionen `newCPoly`, `delCPoly`, `getCPolyDegree`, `getCPolyCoefficient` und `setCPolyCoefficient`. Speichern Sie den Source-Code unter `cpoly.c` in das Verzeichnis `serie11`.

Aufgabe 11.6. 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 11.5. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem zwei Polynome p, q eingelesen und die Summe $r = p + q$ ausgegeben werden. Speichern Sie den Source-Code unter `addcpoly.c` in das Verzeichnis `serie11`.

Aufgabe 11.7. Schreiben Sie eine Struktur `CVector`, zur Speicherung von Vektoren mit komplexwertigen Koeffizienten. Benutzen Sie hierzu den Strukturdatentyp `cdouble` aus Aufgabe 11.2. Ferner schreibe man die zugehörigen Funktionen `newCVector`, `delCVector`, `getCVectorLength`, `getCVectorEntry`, `setCVectorEntry`. Speichern Sie den Source-Code unter `cvector.c` in das Verzeichnis `serie11`.

Aufgabe 11.8. Schreiben Sie eine Funktion `CVectorVector`, die das Skalarprodukt $x \cdot y := \sum_{j=1}^{n} x_j \overline{y_j}$ zweier komplexwertiger Vektoren $x, y \in \mathbb{C}^n$ berechnet. Benutzen Sie hierzu den Strukturdatentyp `CVector` aus Aufgabe 11.7. Schreiben Sie ferner ein aufrufendes Hauptprogramm, indem die Vektoren x, y eingelesen und der Wert $x \cdot y \in \mathbb{C}$ ausgegeben werden. Speichern Sie den Source-Code unter `CVectorVector.c` in das Verzeichnis `serie11`.