Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (2 Punkte):
Aufgabe 2 (2 Punkte):
Aufgabe 3 (1 Punkte):
Aufgabe 4 (1 Punkte):
Aufgabe 5 (2 Punkte):
Aufgabe 6 (3 Punkte):
Aufgabe 7 (3 Punkte):
Aufgabe 8 (4 Punkte):
Aufgabe 9 (4 Punkte):
Aufgabe 10 (8 Punkte):

Gesamtpunktzahl:

Schriftlicher Nachtest zu C (90 Minuten)
VU Einführung ins Programmieren für TM (SS 2009)

01. Oktober 2009

Aufgabe 1 (2 Punkte). Schreiben Sie einen Struktur-Datentyp `Cdouble` zur Speicherung von komplexen Zahlen $x \in \mathbb{C}$. In der Struktur sollen der Realteil `real` und der Imaginärteil `imag` gespeichert werden. Diese Struktur soll auch in den übrigen Aufgaben des Tests verwendet werden.

Lösung zu Aufgabe 1.

Aufgabe 2 (2 Punkte). Schreiben Sie eine Funktion `newCdouble`, die eine neue komplexe Zahl allokiert und initialisiert.

Lösung zu Aufgabe 2.

Aufgabe 3 (1 Punkt). Schreiben Sie eine Funktion `setCimag`, die den Imaginärteil einer komplexen Zahl zuweist.

Lösung zu Aufgabe 3.

Aufgabe 4 (1 Punkt). Schreiben Sie eine Funktion `getCimag`, die den Imaginärteil einer komplexen Zahl zurückgibt.

Lösung zu Aufgabe 4.

Hinweis: In den folgenden Aufgaben dürfen Sie — zusätzlich zu den von Ihnen programmierten — folgende Funktionen verwenden:

- `void setCreal(Cdouble* x, double real)`
- `double getCreal(Cdouble* x)`

Aufgabe 5 (2 Punkte). Schreiben Sie eine Funktion `Cabs`, die die euklidische Länge

$$|x| := \sqrt{a^2 + b^2}$$

einer komplexen Zahl $x = a + bi \in \mathbb{C}$ mit $a, b \in \mathbb{R}$ zurückgibt.

Lösung zu Aufgabe 5.

Aufgabe 6 (3 Punkte). Schreiben Sie eine Funktion `Cdivide`, die für zwei komplexe Zahlen $x, y \in \mathbb{C}$ mit $y \neq 0$ den Quotienten x/y berechnet und das Ergebnis entsprechend zurückgibt.

Aufgabe 7 (3 Punkte). Schreiben Sie einen Strukturdatentyp `Cvector` zur Speicherung von Vektoren $x \in \mathbb{C}^n$ mit komplexwertigen Koeffizienten x_j. In der Struktur sollen neben der Länge $n \in \mathbb{N}$ auch die n Koeffizienten $x_j \in \mathbb{C}$ gespeichert werden. Diese Struktur soll auch in den übrigen Aufgaben des Tests verwendet werden.

Lösung zu Aufgabe 7.

Aufgabe 8 (4 Punkte). Schreiben Sie eine Funktion `newCvector`, die für gegebene Länge $n \in \mathbb{N}$ einen Vektor $x \in \mathbb{C}^n$ allokiert und initialisiert.

Lösung zu Aufgabe 8.

Aufgabe 9 (4 Punkte). Schreiben Sie eine Funktion `delCvector`, die den Speicher für einen dynamisch allokierten Vektor $x \in \mathbb{C}^n$ freigibt und den `NULL`-Pointer zurückgibt.

Lösung zu Aufgabe 9.

Hinweis: In der abschließenden Aufgabe dürfen Sie — zusätzlich zu den von Ihnen programmierten — folgende Funktionen verwenden:

- `int getCvectorLength(Cvector* x)`
- `Cdouble* getCvectorCoeff(Cvector* x, int j)`

Gehen Sie davon aus, dass die Vektoren wie in der Mathematik üblich indiziert werden, d.h. $x = (x_1, \ldots, x_n) \in \mathbb{C}^n$.

Aufgabe 10 (8 Punkte). Schreiben Sie eine Funktion `maxCvector`, die für einen gegebenen Vektor $x \in \mathbb{C}^n$ den kleinsten Index j mit

$$|x_j| = \max_{k=1,\ldots,n} |x_k|$$

zurückgibt. Was müsste man ändern, um den größten solchen Index j zu erhalten?