Dirk Praetorius,
Marcus Page

Sommersemester 2011
13. März 2011

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 2

Aufgabe 2.1*. Schreiben Sie eine `void`-Funktion `teiler`, die für eine gegebene Zahl $x \in \mathbb{N} := \{1, 2, 3, \dots\}$ ausgibt, ob diese durch 2, durch 3 oder durch 6 teilbar ist. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das den Integer x einliest und `teiler` aufruft. Speichern Sie den Source-Code unter `teiler.c` in das Verzeichnis `serie02`.

Aufgabe 2.2*. Schreiben Sie eine `void`-Funktion `quadrant`, die für einen Punkt $(x, y) \in \mathbb{R}^2$ ausgibt, ob (x, y) auf einer der Achsen des Koordinatensystems liegt. Falls nicht, soll ausgegeben werden, in welchem Quadranten (x, y) liegt. Schreiben Sie ferner ein Hauptprogramm, in dem $x, y \in \mathbb{R}$ eingelesen werden. Speichern Sie den Source-Code unter `quadrant.c` in das Verzeichnis `serie02`.

Aufgabe 2.3*. Schreiben Sie eine Funktion `vecmult`, der zwei Vektoren $x \in \mathbb{R}^a$ und $y \in \mathbb{R}^b$ und deren Längen a und b übergeben werden. Sind beide Vektoren gleich lang, so soll die Funktion das Produkt

$$x^T y = \sum_{j=1}^{a} x_j y_j$$

ausgeben und falls $a \neq b$ eine entsprechende Meldung. Schreiben Sie ausserdem ein aufrufendes Hauptprogramm, in dem die Vektoren von der Tastatur eingelesen werden. Die Längen sollen im Hauptprogramm Konstanten sein, die Funktion `vecmult` ist jedoch für allgemeine Längen zu implementieren. Speichern Sie den Source-Code unter `vecmult.c` in das Verzeichnis `serie02`.

Aufgabe 2.4*. Schreiben Sie eine Funktion `norm`, welche das Quadrat der ℓ^2-Norm, d.h.

$$\|x\|_2^2 = \sum_{j=1}^{n} x_i^2$$

eines Vektors x der Länge n zurückgibt. Ein aufrufendes Programm soll ferner den Vektor x einlesen. Die Größe n soll im Hauptprogramm wieder konstant sein. Speichern Sie den Source-Code unter `norm.c` in das Verzeichnis `serie02`.
Tip: Verwenden Sie Aufgabe 2.3.

Aufgabe 2.5. Schreiben Sie zusätzlich zu Ihrer Funktion `evalpol` aus Aufgabe 1.8 eine Funktion `evalpol2`, welche mathematisch das gleiche realisiert, jedoch mit dem `pow`-Befehl aus der mathematischen Bibliothek arbeitet. Achten Sie darauf, die Bibliothek beim kompilieren entsprechend zu verlinken. Ein aufrufendes Hauptprogramm soll nun Koeffizienten und einen Auswertungspunkt x einlesen und die korrespondierenden Polynome mittels `evalpol` und `evalpol2` auswerten. Die beiden Ergebnisse sollen schliesslich auf Gleichheit überprüft, und das Ergebnis entsprechend kommentiert werden. Speichern Sie den Source-Code unter `evalpol2.c` in das Verzeichnis `serie02`.

Aufgabe 2.6. Schreiben Sie eine Funktion `pol2cart`, die gegebene Polarkoordinaten (r, φ) in kartesische Koordinaten umrechnet. Verwenden Sie auch hierbei die mathematische Bibliothek. Schreiben Sie ferner ein aufrufendes Hauptprogramm, welches die Größen r und φ an `pol2cart` übergibt. Speichern Sie den Source-Code unter `pol2cart.c` in das Verzeichnis `serie02`.

Aufgabe 2.7. Schreiben Sie eine Funktion `matmult`, die eine Matrix-Vektor-Multiplikation realisiert. Die Matrix $M \in \mathbb{R}^{a \times b}$ und der Vektor $x \in \mathbb{R}^c$ sollen hierbei inklusive der entsprechenden Grössen a, b und c übergeben werden. Das Resultat $y \in \mathbb{R}^a$ soll schliesslich ausgegeben werden. Die Größen a, b und c sollen wieder Konstanten im Hauptprogramm sein. Speichern Sie den Source-Code unter `matmult.c` in das Verzeichnis `serie02`.

Aufgabe 2.8. Schreiben Sie eine Funktion `matmult2`, die eine Matrix-Matrix-Multiplikation realisiert. Orientieren Sie sich hierbei an Aufgabe 2.7. Testen Sie Ihren Code anhand einiger Beispiele. Finden Sie eine Möglichkeit, die Produktmatrix sinnvoll auszugeben. Speichern Sie den Source-Code unter `matmult2.c` in das Verzeichnis `serie02`.

Aufgabe 2.9. Schreiben Sie eine Funktion `findme`, die einen Vektor $v \in \mathbb{N}^a$ nach einer Zahl $x \in \mathbb{N}$ durchsucht. Die Zahl x und der Vektor v soll von einem aufrufenden Hauptprogramm übergeben werden. Die Funktion soll dann den höchsten Index zurückgeben, an dem x steht. Falls x überhaupt nicht vorkommt, so soll (-1) zurückgegeben werden. Achten Sie darauf `findme` für beliebige Längen $a \in \mathbb{N}$ zu implementieren und übergeben Sie diesen Wert entsprechend. Speichern Sie den Source-Code unter `findme.c` in das Verzeichnis `serie02`.

Aufgabe 2.10. Schreiben Sie eine Funktion `sum` die für gegebenes $n \in \mathbb{N}$ die Summe $\sum_{j=1}^{n}(j/2)$ berechnet und zurückgibt. Realisieren Sie diese Summe dabei direkt und nicht in der Form $\frac{1}{2}\sum_{j=1}^{n} j$. Was ist zu beachten? Schreiben Sie ferner ein Hauptprogramm, das den Wert n von der Tastatur einliest und das Ergebnis `sum` am Bildschirm ausgibt. Speichern Sie den Source-Code unter `sum.c` in das Verzeichnis `serie02`.