Dirk Praetorius, Samuel Ferraz-Leite
Christoph Überhuber

Wintersemester 2008/09
29. Oktober 2008

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 3

Die Aufgaben mit Stern (*) sind bis zur Übung in der kommenden Woche vorzubereiten und werden dort abgeprüft. Die übrigen Aufgaben dienen nur Ihrer Übung und uns als zusätzliche Grundlage für den Prüfungsstoff in den schriftlichen Tests. Kopieren Sie bitte den Source-Code auf Ihren Account auf der `lva.student.tuwien.ac.at` *in ein Unterverzeichnis* `serie03`*. Überprüfen Sie bitte vor der Übung, ob Ihre Source-Codes mit dem* `gcc` *compiliert werden können. In den folgenden Übungsaufgaben sollen* ***Zählschleifen*** *sowie* ***statische Vektoren und Matrizen*** *geübt werden.*

Aufgabe 21*. Schreiben Sie eine Funktion `minmaxmean`, die von einem gegebenem Vektor $x \in \mathbb{R}^n$ das Minimum $\min_{j=1}^n x_j$, das Maximum $\max_{j=1}^n x_j$ und den Mittelwert $\frac{1}{n}\sum_{j=1}^n x_j$ berechnet und zurückgibt. Ferner schreibe man ein aufrufendes Hauptprogramm, das den Vektor x einliest und die von `minmaxmean` berechneten Kenngrößen ausgibt. Die Länge $n \in \mathbb{N}$ des Vektors soll eine Konstante im Hauptprogramm sein, die Funktion `minmaxmean` ist für beliebige Länge n zu programmieren. Speichern Sie den Source-Code unter `minmaxmean.c` in das Verzeichnis `serie03`.

Aufgabe 22*. Ein Tripel $(x, y, z) \in \mathbb{N}^3$ natürlicher Zahlen heißt *pythagoräisches Zahlentripel*, falls $x^2 + y^2 = z^2$ gilt. Das wohl bekannteste Beispiel ist $(3, 4, 5)$. Offensichtlich gelten $z > \max\{x, y\}$ sowie $x \neq y$ und ohne Beschränkung der Allgemeinheit ferner $x < y$. Man schreibe eine Prozedur `pythagoras`, die zu gegebener Schranke $n \in \mathbb{N}$ alle pythagoräischen Zahlentripel mit $x < y < z \leq n$ bestimmt und ausgibt. Ferner schreibe man ein aufrufendes Hauptprogramm, in dem die Schranke n eingelesen und `pythagoras` aufgerufen wird. Speichern Sie den Source-Code unter `pythagoras.c` in das Verzeichnis `serie03`.

Aufgabe 23*. Was sind die Bestandteil einer Gleitpunktzahl des Zahlensystems $\mathbb{F}(b, p, e_{\min}, e_{\max})$? Wie groß ist die kleinste positive normalisierte Gleitpunktzahl in $\mathbb{F}(16, 6, -64, 63)$? Was ist ein implizites erstes Bit? Kann man bei dezimalen Gleitpunktzahlen (mit Basis 10) ein implizites erstes Bit verwenden?

Aufgabe 24*. Die Gleitpunkt-Arithmetik ist nicht assoziativ. Das numerische Ergebnis der Summe $\sum_{j=0}^n x^j/j!$ hängt daher von der Summationsreihenfolge ab. Man schreibe Funktionen `forward` und `backward`, die diese Summe auf zwei Weisen berechnen: `forward` entspricht Vorwärtssummation $j = 0, \ldots, n$, `backward` entspricht Rückwärtssummation $j = n, \ldots, 0$. Welche Variante wird besser sein – zumindest für positives x und großes n? Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem die Schranke $n \in \mathbb{N}$ eingelesen und die Ergebnisse von `forward` bzw. `backward` ausgegeben werden. Speichern Sie den Source-Code unter `forwardbackward.c` in das Verzeichnis `serie03`.

Aufgabe 25. Gegeben sei ein Polynom $p(x) = \sum_{j=0}^{n} a_j x^j$ in Form seines Koeffizientenvektors $a = (a_0, \ldots, a_n) \in \mathbb{R}^{n+1}$. Schreiben Sie eine Funktion `evalpolynomial`, die für gegebenen Koeffizientenvektor a und Auswertungspunkt x den Funktionswert $p(x)$ berechnet. Die Funktion `pow` zur Berechnung von x^j soll *nicht* verwendet werden. Schreiben Sie eine Funktion, die möglichst nur *eine* Schleife verwendet. Der Grad $n \in \mathbb{N}$ des Polynoms soll eine Konstante im Hauptprogramm sein, die Funktion `evalpolynomial` soll aber beliebigen Grad zulassen. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem die Koeffizienten a_j sowie der Auswertungspunkt x eingelesen werden und $p(x)$ ausgegeben wird.

Aufgabe 26. Die Fibonacci-Folge ist definiert durch $x_0 := 0$, $x_1 := 1$ und $x_{n+1} := x_n + x_{n-1}$. Man schreibe eine Funktion `fibonacci`, die zu gegebenem Index n das Folgenglied x_n zurückgibt. Ferner schreibe man ein aufrufendes Hauptprogramm, das den Index n einliest und x_n ausgibt.

Aufgabe 27. Eine Matrix $A \in \mathbb{R}^{n \times n}$ ist symmetrisch, falls $A_{jk} = A_{kj}$ für alle $j, k = 1, \ldots, n$ gilt. Schreiben Sie eine Funktion `issymmetric`, die eine Matrix A auf Symmetrie überprüft (Rückgabewert `1` bei Symmetrie und `0` bei Nicht-Symmetrie). Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem A eingelesen wird und ausgegeben wird, ob A symmetrisch ist oder nicht. Die Dimension $n \in \mathbb{N}$ der Matrix soll eine Konstante im Hauptprogramm, aber ein Parameter der Funktion `issymmetric` sein.

Aufgabe 28. Für eine Matrix $A \in \mathbb{R}^{m \times n}$ ist die Spaltensummennorm durch

$$\|A\| = \max_{j=1,\ldots,n} \sum_{i=1}^{m} |A_{ij}|$$

gegeben. Schreiben Sie eine Funktion `spaltensummennorm`, die die Spaltensummennorm einer Matrix A berechnet. Schreiben Sie ein aufrufendes Hauptprogramm, in dem A eingelesen und $\|A\|$ ausgegeben wird. Die Dimensionen $m, n \in \mathbb{N}$ der Matrix sollen Konstanten im Hauptprogramm, aber Parameter der Funktion `spaltensummenorm` sein.

Aufgabe 29. Man schreibe eine Funktion `prim`, die überprüft, ob eine natürliche Zahl $n \in \mathbb{N}$ eine Primzahl ist (Rückgabewert 1) oder nicht (Rückgabewert 0). Ferner schreibe man ein aufrufendes Hauptprogramm, das den Wert n einliest und ausgibt, ob es sich um eine Primzahl handelt.

Aufgabe 30. Um die Summe $\sum_{j=1}^{n} (-1)^j / j$ zu berechnen, ist es numerisch günstig, zunächst die negativen und die positiven Beiträge getrennt zu summieren und erst abschließend beide Teilsummen zu addieren. Warum? Man schreibe eine Funktion `sum`, die dieses Vorgehen realisiert. Ferner schreibe man ein Hauptprogramm, das n einliest und $\sum_{j=1}^{n} (-1)^j / j$ ausgibt.