Dirk Praetorius, Samuel Ferraz-Leite
Christoph Überhuber

Wintersemester 2009/10
16. Dezember 2009

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 9

Die Aufgaben mit Stern (*) sind bis zur Übung in der kommenden Woche vorzubereiten. Kopieren Sie die Source-Codes vor der Übung auf Ihren Account auf der `lva.student.tuwien.ac.at` *und überprüfen Sie, ob diese von* `matlab` *auf der* `lva.student.tuwien.ac.at` *interpretiert werden können. In den folgenden Aufgaben sollen* **Bedingungsschleifen, Function-Handles** *und* **rekursive Funktionen** *geübt werden.*

Aufgabe 9.1*. Schreiben Sie eine Funktion `merge`, die zwei bereits aufsteigend sortierte Felder a und b so vereinigt, dass das resultierende Feld c ebenfalls aufsteigend sortiert ist, z.B. soll also Aufruf mit $a = [1, 3, 3, 4, 7]$ und $b = [1, 2, 3, 8]$ als Ergebnis $c = [1, 1, 2, 3, 3, 3, 4, 7, 8]$ liefern. Speichern Sie den Source-Code im Verzeichnis `serie09` unter `merge.m`.

Aufgabe 9.2*. Schreiben Sie eine rekursive Funktion `mergesort`, die ein Feld a aufsteigend sortiert und das sortierte Feld zurückgibt. Gehen Sie dabei nach folgender Strategie vor:

- Hat a Länge ≤ 2, so wird das Feld a explizit sortiert.
- Hat a Länge > 2, halbiert man a in zwei Teilfelder a_1 und a_2. Man ruft rekursiv `mergesort` für a_1 und a_2 auf und vereinige die sortierten Teilfelder mittels `merge` aus Aufgabe 9.1.

Machen Sie sich das Vorgehen anhand des einfachen Beispiels $a = [1, 3, 5, 2, 7, 1, 1, 3]$ klar. Speichern Sie den Source-Code im Verzeichnis `serie09` unter `mergesort.m`.

Aufgabe 9.3*. Die Fibonacci-Folge ist definiert durch $x_0 := 0, x_1 := 1$ und $x_{n+1} = x_n + x_{n-1}$. Schreiben Sie eine rekursive Funktion `fibonacciRek`, die zu gegebenem Index n das Folgenglied x_n zurückgibt. Schreiben Sie weiters eine nicht rekursive Funktion `fibonacci` die dasselbe leistet, wobei die Berechnung aber über geeignete Schleifen realisiert wird. Welche der beiden Funktionen ist effizienter und warum? Speichern Sie die Source-Codes im Verzeichnis `serie09`.

Aufgabe 9.4*. Alternativ zum Bisektionsverfahren aus der Vorlesung kann man zur Berechnung einer Nullstelle einer Funktion $f : [a, b] \to \mathbb{R}$ das *Newton-Verfahren* verwenden. Ausgehend von einem Startwert x_0 definiert man induktiv eine Folge (x_n) wie folgt: Zu gegebenem x_k sei x_{k+1} die Nullstelle der Tangente an den Graphen von f im Punkt $(x_k, f(x_k))$, d.h. $x = x_{k+1}$ erfüllt $0 = f(x_k) + f'(x_k)(x - x_k)$. Auflösen nach x zeigt

$$x_{k+1} = x_k - f(x_k)/f'(x_k).$$

Realisieren Sie das Newton-Verfahren in einer Funktion `newton(f,fprime,x0,tau)`, wobei die Iteration abgebrochen wird, falls entweder

$$|f'(x_n)| \leq \tau$$

oder

$$|f(x_n)| \leq \tau \quad \text{und} \quad |x_n - x_{n-1}| \leq \begin{cases} \tau & \text{für } |x_n| \leq \tau, \\ \tau|x_n| & \text{sonst} \end{cases}$$

gilt. In jedem Fall werde $\widetilde{z} := x_n$ als Approximation der gesuchten Nullstelle zurückgegeben, wobei im ersten Fall zusätzlich eine Warnung ausgegeben werden soll. — Neben x_n sollen die Folgen $(x_0, \dots, x_n)$ der Nullstellen und der dazugehörigen Funktionswerte zurückgegeben werden. Testen Sie Ihre Implementierung mit der Funktion $f(x) = x^2 + e^x - 2$. Speichern Sie den Source-Code im Verzeichnis `serie09` unter `newton.m`.

Aufgabe 9.5. Für $x > 0$ konvergiert die Folge

$$x_1 := \frac{1}{2}(1+x), \quad x_{n+1} := \frac{1}{2}\Big(x_n + \frac{x}{x_n}\Big) \quad \text{für } n \geq 1$$

gegen $\sqrt{x}$. Schreiben Sie eine Funktion `sqrt_`, die für gegebene $x > 0$ und $\varepsilon > 0$ als Ergebnis das erste Folgenglied $y = x_n$ zurückgibt, für das gilt

$$\frac{|x_n - x_{n+1}|}{|x_n|} \leq \varepsilon \quad \text{oder} \quad |x_n| \leq \varepsilon.$$

Welcher Zusammenhang besteht mit dem Newton-Verfahren aus Aufgabe 9.4?

Aufgabe 9.6. Für eine differenzierbare Funktion $f : [a, b] \to \mathbb{R}$ kann man die Ableitung $f'(x)$ in einem festen Punkt $x \in \mathbb{R}$ durch den einseitigen Differenzenquotienten

$$\Phi(h) := \frac{f(x+h) - f(x)}{h} \quad \text{für } h > 0$$

approximieren. Nach Taylor-Formel gilt für $f \in C^2(\mathbb{R})$ die Fehlerabschätzung $|f'(x) - \Phi(h)| = \mathcal{O}(h)$. Beweisen Sie diese Aussage mathematisch! Schreiben Sie eine Funktion `diff(f,x,h0,tau)`, die für $h_n := 2^{-n} h_0$ die Folge der $\Phi(h_n)$ berechnet, bis gilt

$$|\Phi(h_n) - \Phi(h_{n+1})| \leq \begin{cases} \tau & \text{für } |\Phi(h_n)| \leq \tau, \\ \tau\,|\Phi(h_n)| & \text{anderenfalls.} \end{cases}$$

Die Funktion liefere $y = \Phi(h_n)$ zurück. — Alternativ soll die vollständige Folge $(\Phi(h_0), \dots, \Phi(h_n))$ der Iterierten zurückgegeben werden.

Aufgabe 9.7. Das Newton-Verfahren aus Aufgabe 9.4 benötigt neben der Funktion `f` auch eine Funktion `fprime`, die die Ableitung f' der Funktion f auswertet. Alternativ kann man $f'(x_k)$ durch den Differenzenquotienten $\Phi_h(x_k)$ aus Aufgabe 9.6 ersetzen. Realisieren Sie dieses Vorgehen indem Sie eine Funktion `newton(f,x0,h0,tau)` schreiben, die zur Approximation der Abeitung $f'(x_k)$ das Ergebnis von `diff(f,xk,h0,tau)` verwendet.

Aufgabe 9.8. Schreiben Sie eine rekursive Funktion `binomial`, die den Binomialkoeffizienten $\binom{n}{k}$ berechnet. Dazu verwende man das Additionstheorem $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. Schreiben Sie ferner eine Funktion `binomial2`, die den Binomialkoeffizienten mittels $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ berechnet. Dazu ist eine rekursive Funktion `faktorielle` zu entwickeln, die zu gegebenem $n \in \mathbb{N}$ die Faktorielle $n!$ berechnet.

Aufgabe 9.9. Schreiben Sie eine rekursive Funktion `detlaplace`, die die Determinante $\det(A)$ einer Matrix $A \in \mathbb{R}^{n \times n}$ mit Hilfe des Laplaceschen Entwicklungssatzes berechnet. Sie können Ihre Funktion mit der MATLAB-Funktion `det` verifizieren.

Aufgabe 9.10. Schreiben Sie eine (rekursive) Funktion `papierschnitt`, die alle Möglichkeiten visualisiert, wie ein Papierbogen der ganzzahligen Länge `laenge` in Papierbahnen der Länge 1 und 2 geschnitten werden kann. — D.h. man stelle eine natürliche Zahl $n =$ `laenge` auf alle möglichen Weisen als Summe $n = \sum_{j=1}^{k} \sigma_j$ mit Summanden $\sigma_j \in \{1, 2\}$ dar. Dabei soll die Reihenfolge beachtet werden. Für `laenge=4` gibt es beispielsweise 5 Möglichkeiten:

- $4 = 2 + 2$
- $4 = 2 + 1 + 1$
- $4 = 1 + 2 + 1$
- $4 = 1 + 1 + 2$
- $4 = 1 + 1 + 1 + 1$