Dirk Praetorius, Samuel Ferraz-Leite
Christoph Überhuber

Wintersemester 2010/11
01. Dezember 2010

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 7

Aufgabe 7.1*. Für $p \in [1, \infty)$ ist die ℓ_p-Norm auf $\mathbb{R}^n$ definiert durch

$$\|x\|_p := \Big(\sum_{j=1}^{n} |x_j|^p \Big)^{1/p}.$$

Schreiben Sie eine Funktion `pnorm(x,p)`, die einen Vektor $x \in \mathbb{R}^n$ sowie $p \in [1, \infty)$ übernimmt und $\|x\|_p$ zurückgibt. Testen Sie Ihr Programm mit verschiedenen Werten für p bei festem Vektor x. Was beobachten Sie für $p \to \infty$? Speichern Sie den Source-Code unter `pnorm.m` in das Verzeichnis `serie07`.

Aufgabe 7.2*. Schreiben Sie eine Funktion `dreieck`, die für gegebene Seitenlängen $a, b, c \in \mathbb{R}$ mit $a, b, c \geq 0$ feststellt, ob es sich bei dem zugehörigen Dreieck um ein allgemeines, gleichschenkeliges, gleichseitiges, rechtwinkeliges, eindimensional „entartetes" oder um ein „unmögliches" Dreieck handelt. Speichern Sie den Source-Code unter `dreieck.m` in das Verzeichnis `serie07`.

Aufgabe 7.3*. Schreiben Sie eine Funktion `vielfache(k,nmax)`, die alle ganzzahligen Vielfachen der Zahl $k \in \mathbb{N}$, die $\leq n_{\max} \in \mathbb{N}$ sind, am Bildschirm ausgibt. Die Ausgabe erfolge zeilenweise in der Form

```
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
```

beispielsweise für den Fall $k = 5$ und $n_{\max} = 19$. Speichern Sie den Source-Code unter `vielfache.m` in das Verzeichnis `serie07`.

Aufgabe 7.4*. Schreiben Sie eine Funktion `binomial`, die mittels *einer* geeigneten Schleife den Binomialkoeffizienten $\binom{n}{k}$ berechnet. Dazu realisiere man die gekürzte Form

$$\binom{n}{k} = \frac{n \cdot (n-1) \cdots (n-k+1)}{k \cdot (k-1) \cdots 1}.$$

Schreiben Sie weiters eine Lösung mit *zwei* Schleifen, bei der Zähler und Nenner getrennt berechnet werden. Speichern Sie den Source-Code unter `binomial.m` in das Verzeichnis `serie07`. Welche der zwei Implementierungen ist klüger und warum?

Aufgabe 7.5. Die Quotientenfolge $(a_{n+1}/a_n)_{n \in \mathbb{N}}$ zur Fibonacci-Folge $(a_n)_{n \in \mathbb{N}}$,

$$a_0 := 1, \quad a_1 := 1, \quad a_n := a_{n-1} + a_{n-2} \quad \text{für } n \geq 2,$$

konvergiert gegen den goldenen Schnitt $(1 + \sqrt{5})/2$. Insbesondere konvergiert die Differenz

$$b_n := \frac{a_{n+1}}{a_n} - \frac{a_n}{a_{n-1}}$$

gegen Null. Schreiben Sie eine Funktion `cauchy`, die zu gegebenem $k \in \mathbb{N}$ die kleinste Zahl $n \in \mathbb{N}$ mit $|b_n| \leq 1/k$ zurückgibt.

Aufgabe 7.6. Gegeben sei ein Kreis in Form seines Mittelpunkts $(x, y) \in \mathbb{R}^2$ und seines Radius $r > 0$. Gegeben sei ferner ein Punkt $(u, v) \in \mathbb{R}^2$. Schreiben Sie eine Funktion `locate`, die zurückgibt, ob der Punkt (u, v) im Kreis (Rückgabe `-1`), auf der Kreislinie (Rückgabe `0`) oder außerhalb des Kreises (Rückgabe `1`) liegt.

Aufgabe 7.7. Die k-te Ableitung $p^{(k)}$ eines Polynoms p ist wieder ein Polynom. Schreiben Sie eine Funktion `differentiatePolynomial`, die zu gegebenem p und $k \in \mathbb{N}$ die Ableitung $p^{(k)}$ berechnet. Das Ergebnispolynom $p^{(k)}$ soll minimalen Grad $n \in \mathbb{N}$ haben, d.h. es gilt $a_n \neq 0$ für den höchsten Koeffizienten von $p^{(k)}$. Dabei sollen Polyynome in Form der Zeilenvektoren ihrer Koeffizienten gespeichert werden.

Aufgabe 7.8. Schreiben Sie eine Funktion `evalDiffPoly`, die für ein gegebenes Polynom p, eine Ableitungsordnung $k \in \mathbb{N}_0$ und einen Punkt $x \in \mathbb{R}$ den Funktionswert $p^{(k)}(x)$ zurückgibt. Dabei soll das Polynom $p^{(k)}$ nicht explizit gebildet und gespeichert werden.

Aufgabe 7.9. Schreiben Sie eine Funktion `exponential`, die den Funktionswert $\exp(x)$ approximativ berechnet: Dazu berechnen Sie die Partialsumme

$$S_N(x) := \sum_{j=0}^{N} \frac{x^j}{j!},$$

wobei die Summationsgrenze $N \in \mathbb{N}$ durch das Kriterium

$$\left|\frac{x^{N+1}}{(N+1)!}\right| \leq \left|\frac{x^N}{N!}\right| \leq \varepsilon$$

für eine gegebene Toleranz $\varepsilon > 0$ bestimmt werde. Intern realisiere man die Berechnung der Summationsglieder $x^j/j!$ möglichst rechenökonomisch. Vergleichen Sie den Fehler $|S_N(x) - \exp(x)|$ für verschiedene Wahlen von $\varepsilon > 0$ und Auswertungspunkten $x \in \mathbb{R}$.

Aufgabe 7.10. Gegeben Sei ein sortierter int Vektor x der Länge n. Schreiben Sie eine Funktion `find(x,y)`, die einen Index i zurückgibt, für den x(i)=y gilt. Falls y nicht in x vorkommt, werde -1 zurückgegeben. Naive Realisierung führt auf Aufwand n im worst case, d.h. man durchsucht den Vektor von vorne nach hinten und das gesuchte y steht entweder an der letzten Stelle in x bzw. kommt überhaupt nicht in x vor. Wie kann man den Bisektionsalgorithmus aus der Vorlesung geeignet modifizieren, um einen Algorithmus mit worst case Aufwand $\log(n)$ zu erhalten?