Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
11.-13.11.2015

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 5

Aufgabe 5.1. 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`, die einen Vektor $x \in \mathbb{R}^n$, dessen Länge n sowie $p \in [1, \infty)$ übernimmt und $\|x\|_p$ zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem x und p eingelesen werden und $\|x\|_p$ ausgegeben wird. Die Dimension $n \in \mathbb{N}$ soll eine Konstante im Hauptprogramm sein, die Funktion `pnorm` soll aber beliebige Dimension zulassen. 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.c` in das Verzeichnis `serie05`.

Aufgabe 5.2. Schreiben Sie eine Funktion `maxabs`, die von einem gegebenem Vektor $x \in \mathbb{R}^n$ das erste Element x_j mit maximalem Betrag berechnet und zurückgibt, d.h. $|x_j| = \max\{|x_i| : i = 1, \dots, n\}$ und falls $|x_i| = |x_j|$ dann gilt $i \geq j$. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das den Vektor x einliest und das Ergebnis von `maxabs` ausgibt. Der Vektor x soll dabei mittels statischem Array realisiert werden. Die Länge des Vektors soll eine Konstante im Hauptprogramm sein, die Funktion `maxabs` ist aber für beliebige Länge zu implementieren. Speichern Sie den Source-Code unter `maxabs.c` in das Verzeichnis `serie05`.

Aufgabe 5.3. Schreiben Sie die Funktion

```
nk = binomial(n,k,type)
```

die den Binomialkoeffizienten $\binom{n}{k}$ berechnen. Dies läßt sich auf verschiedene Weisen realisieren:

- direkt in der Form $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ unter Verwendung einer Funktion für die Faktorielle (`type=1`),
- in gekürzter Form $\binom{n}{k} = \frac{n \cdot (n-1) \cdots (n-k+1)}{k \cdot (k-1) \cdots 1}$ mittels geeigneter Schleifen (`type=2`),
- mittels einer rekursiven Funktion, die das Additionstheorem $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$ benutzt (`type=3`).

Realisieren Sie alle 3 Varianten und zusätzlich ein Hauptprogramm, das die Zahlen n und k über die Tastatur einliest und den Binomialkoeffizien berechnet und ausgibt. Speichern Sie den Source-Code unter `binomial.c` in das Verzeichnis `serie05`.

Aufgabe 5.4. Schreiben Sie eine *nicht-rekursive* Funktion `power`, die für gegebene reelle Zahlen $x > 1$ und $C > 0$ die kleinste Zahl $n \in \mathbb{N}$ berechnet mit $x^n > C$. Dabei soll die Funktion `log` nicht verwendet werden. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem x und C eingelesen werden und n ausgegeben wird. Speichern Sie den Source-Code unter `power.c` in das Verzeichnis `serie05`.

Aufgabe 5.5. 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 *nicht-rekursive* Funktion `cauchy`, die zu gegebenem $k \in \mathbb{N}$ die kleinste Zahl $n \in \mathbb{N}$ mit $|b_n| \leq 1/k$ zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das die Zahl $k \in \mathbb{N}$ einliest und den zugehörigen Index $n \in \mathbb{N}$ ausgibt. Speichern Sie den Source-Code unter `goldenerSchnitt.c` in das Verzeichnis `serie05`.

Aufgabe 5.6. Die Sinus-Funktion hat die Reihendarstellung

$$\sin(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}.$$

Wir betrachten die Partialsummen

$$S_n(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k+1}}{(2k+1)!}.$$

Schreiben Sie eine *nicht-rekursive* Funktion `sin_`, die für gegebene $x \in \mathbb{R}$ und $\varepsilon > 0$ den Wert $S_n(x)$ zurückliefert, sobald

$$|S_n(x) - S_{n-1}(x)|/|S_n(x)| \leq \varepsilon \quad \text{oder} \quad |S_n(x)| \leq \varepsilon$$

gilt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem $x \in \mathbb{R}$ und $\varepsilon > 0$ eingelesen werden. Neben dem berechneten Wert $S_n(x)$ sollen auch der korrekte Wert $\sin(x)$ und der absolute Fehler $|S_n(x) - \sin(x)|$ ausgegeben werden sowie der relative Fehler $|S_n(x) - \sin(x)|/|\sin(x)|$ im Fall $\sin(x) \neq 0$. Speichern Sie den Source-Code unter `sin.c` in das Verzeichnis `serie05`.

Aufgabe 5.7. 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 *nicht-rekursive* Funktion `sqrt_new`, die für gegebene $x > 0$ und $\tau > 0$ als Ergebnis das erste Folgenglied $y = x_n$ zurückgibt, für das gilt

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

Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem x eingelesen und neben der Approximation x_n von $\sqrt{x}$ auch der exakte Wert sowie der absolute Fehler $|x_n - \sqrt{x}|$ ausgegeben werden. Speichern Sie den Source-Code unter `sqrt_new.c` in das Verzeichnis `serie05`. Vergleichen Sie Ihre Implementierung mit Ihrem Code aus Aufgabe 3.7. Diskutieren Sie Vor- und Nachteile der beiden Implementierungen!

Aufgabe 5.8. Folgendes Programm soll den maximalen Eintrag einer gegebenen Matrix bestimmen. Als Ergebnis wird `5.0000` ausgegeben. Wo liegt der Fehler?

```
#include <stdio.h>

main() {
  double A[2][3] = { {1,2,3},{6,-4,5} };
  double max = A[0][0];

  int j=0, k=0;

  for(j=0; j<2; j=j+1) {
    for(k=1; k<3; k=k+1) {
      if(A[j][k] > max) {
        max = A[j][k];
      }
    }
  }
  printf("Maximum = %f\n",max);
}
```

Beheben Sie den Fehler und erweitern Sie das Programm so, dass auch das Minimum aller Matrixeinträge bestimmt wird. Speichern Sie den Source-Code unter `maxminmatrix.c` in das Verzeichnis `serie05`.