Dirk Praetorius,
Thomas Führer

Sommersemester 2014
31.03.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 5

Aufgabe 5.1. Schreiben Sie eine Funktion `minmaxmean`, die von einem gegebenem Vektor $x \in \mathbb{N}^n$ das Minimum, das Maximum und den Mittelwert $\frac{1}{n}\sum_{j=1}^{n} x_j$ berechnet und geeignet zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das den Vektor $x \in \mathbb{N}^n$ einliest und Minimum, Maximum und Mittelwert 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 `serie05`.

Aufgabe 5.2. Schreiben Sie eine Funktion `kgV(a,b)`, die das kleinste gemeinsame Vielfache zweier natürlicher Zahlen $a, b \in \mathbb{N}$ berechnet. Zur Lösung können Sie entweder die Primfaktoren beider Zahlen berechnen oder den Zusammenhang $a \cdot b = ggT(a,b) \cdot kgV(a,b)$ berücksichtigen. Speichern Sie den Source-Code unter `kgv.c` in das Verzeichnis `serie05`.

Aufgabe 5.3. 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

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

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 $x \in \mathbb{R}$. Speichern Sie den Source-Code unter `exponential.c` in das Verzeichnis `serie05`.

Aufgabe 5.4. 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. 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 5.5. *Bubble-Sort* ist ein ineffizienter, aber kurzer Sortier-Algorithmus: Man vergleicht aufsteigend jedes Element eines Arrays x_j mit seinem Nachfolger x_{j+1} und - falls notwendig - vertauscht die beiden. Nach dem ersten Durchlauf muß zumindest das letzte Element bereits am richtigen Platz sein. Der nächste Durchlauf muß also nur noch bis zur vorletzten Stelle gehen, usw. Wie viele geschachtelte Schleifen braucht dieses Vorgehen? Schreiben Sie eine Funktion `bubblesort`, die ein gegebenes Array $x \in \mathbb{R}^n$ mittels Bubble-Sort aufsteigend sortiert, d.h. $x_1 \leq x_2 \leq \cdots \leq x_n$, und zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das den Vektor x einliest und in sortierter Reihenfolge ausgibt. Die Länge des Vektors soll eine Konstante im Hauptprogramm sein, die Funktion `bubblesort` ist für beliebige Länge n zu programmieren. Speichern Sie den Source-Code unter `bubblesort.c` in das Verzeichnis `serie05`.

Aufgabe 5.6. Gegeben seien die Summen

$$a_N := \sum_{n=0}^{N} \frac{1}{(n+1)^2} \quad \text{und} \quad b_M := a_M^2 = \sum_{m=0}^{M} \sum_{k=0}^{m} \frac{1}{(k+1)^2(m-k+1)^2}.$$

Schreiben Sei ein Programm, welches für verschiedene Werte von N bzw. M die Zeit misst um a_N bzw. b_M zu berechnen. Geben Sie anschließend die Ergebnisse in Form einer Tabelle am Bildschirm aus. Entsprechen die Resultate Ihren Erwartungen? Speichern Sie den Source-Code unter `zeitmessung.c` in das Verzeichnis `serie05`. *Hinweis:* Überlegen Sie sich wie groß der Aufwand bei der Berechnung von a_N bzw. b_M ist.

Aufgabe 5.7. Die Funktion `squareVector` soll alle Einträge eines Vektors $x \in \mathbb{R}^n$ quadrieren, d.h. aus $(-1, 2, 0)$ soll $(1, 4, 0)$ werden. Der Vektor soll dabei als Pointer übergeben werden.

```
#include <stdio.h>

int squareVec(double vec, int n) {
  int j=0;
  for(j=1, j<dim; --j) {
    *vec[j] = &vec[j] * &vec[j];
  }
  return vec;
}

main() {
  double vec[3] = {-1.0,2.0,0.0};
  int j=0;

  squareVec(vec,3);
  for(j=0; j<3; ++j) {
    printf("vec[%d] = %f ",j,vec[j]);
  }
  printf("\n");
}
```

Ändern Sie nur die Funktion `squareVec`, so dass die `main`-Funktion das richtige Ergebnis ausgibt. Wie viele Fehler finden Sie? Welchen Aufwand hat Ihre korrigierte Funktion `squareVec`?

Aufgabe 5.8. Welche Arten von Kommentaren gibt es? Was gibt folgender Code aus und warum?

```
#include <stdio.h>

/*int f(double x) {
    return (int) x;
  }
*/

main() {
  int x = 4;
  int y = 2*x*/* f(0.1)+3
              */1/4;
  // y = 1;
  printf("y = %d\n",y); // Ausgabe
}
```