Dirk Praetorius,
Marcus Page

Wintersemester 2011
04. Nov. 2011

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 4

Aufgabe 4.1*. Für einen gegebenen Vektor $v \in \mathbb{R}^n$ sind ℓ_1-Norm und die *euklidische* Norm definiert durch

$$\|x\|_{\ell_1} := \sum_{i=1}^{n} |x_i|,$$

$$\|x\|_{\ell_2} := \sqrt{\sum_{i=1}^{n} |x_i|^2}.$$

Schreiben Sie eine `void` Funktion `norms`, die zu einem gegebenen Vektor sowohl ℓ_1, als auch euklidische Norm (ℓ_2) berechnet. Die beiden Normen sollen dann im Hauptprogramm verfügbar sein und ausgegeben werden. Insbesondere soll in der Funktion `norms` keine Ausgabe erfolgen. Wie können Sie das realisieren? Speichern Sie den Source-Code unter `norms.c` in das Verzeichnis `serie04`.

Aufgabe 4.2*. Schreiben Sie eine `void`-Funktion `meansum`, die zu einem gegebenen Vektor $x \in \mathbb{R}^n$ sowohl die Summe $s = \sum_{i=1}^{n} x_i$, als auch das harmonische Mittel

$$x_{\text{harm.}} := \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

berechnet. Schreiben Sie außerden ein aufrufendes Hauptprogramm in dem der Vektor x eingelesen und s und $x_{\text{harm.}}$ ausgegeben werden. In der Funktion `meansum` selbst soll keine Ausgabe erfolgen. Speichern Sie den Source-Code unter `meansum.c` in das Verzeichnis `serie04`.

Aufgabe 4.3*. Es gibt viele Möglichkeiten, die Kreiszahl π näherungsweise zu bestimmen. In dieser Übung beschäftigen wir uns mit der *Monte-Carlo-Simulation*. Die Idee hierbei ist es das Einheitsquadrat $[0,1]^2$ mit Zufallszahlen *beregnen* zu lassen. Man wählt hierbei also eine bestimmte Anzahl $N \in \mathbb{N}$ zufälliger Punkte in $[0,1]^2$ und zählt die Anzahl derer die innerhalb des Viertelkreises mit Radius 1 liegen, und diejenigen die sich außerhalb befinden. Der Anteil der innen liegenden Punkte konvergiert nun für $N \to \infty$ gegen $\pi/4$. Schreiben Sie ein Programm welches dieses Vorgehen realisiert. Brechen Sie die Iteration ab, wenn Sie π auf 2 Stellen genau bestimmt haben. Wieviele Iterationen brauchen Sie im Schnitt? Wieviele werden für 3 Nachkommastellen benötigt. Speichern Sie den Source-Code unter `pi.c` in das Verzeichnis `serie04`.

Aufgabe 4.4*. In der Vorlesung haben Sie gelernt, dass es zu jeder Zahl $x \in \mathbb{R}$

- ein Vorzeichen $\sigma \in \{\pm 1\}$,
- Ziffern $a_j \in \{0, 1\}$ und
- einen Exponenten $e \in \mathbb{Z}$

gibt, so dass

$$x = \sigma\Big(\sum_{k=1}^{\infty} a_k 2^{-k}\Big) 2^e$$

gilt (Folie 98). Geben Sie den Beweis dieser Aussage in eigenen Worten wieder.

Aufgabe 4.5. Die Darstellung aus Aufgabe 4.4 ist im Allgemeinen nicht eindeutig. Sie kann aber in einem Gleitkommazahlensystem $\mathbb{F}(2, M, e_{\min}, e_{\max})$ durch feste Mantissenlänge $M \in \mathbb{N}$ und Normierung $a_1 = 1$ eindeutig gemacht werden. Warum? Beweisen Sie diese Aussage. Was wissen Sie über das implizite erste Bit?

Aufgabe 4.6. Was ist eine Schleife? Welche unterschiedlichen Typen von Schleifen gibt es? Erklären Sie die Unterschiede. Was ist der Unterschied und der Zusammenhang zwischen einer Variable und einem Pointer?
Schreiben Sie eine Funktion `swap`, welche die Werte zweier Zahlen a und b vertauscht. Warum funktioniert das folgende Vorgehen nicht

```
void swap(double x, double y)
{
   double tmp;
   tmp = x;
   x = y;
   y = tmp;
}
```

Speichern Sie den Source-Code unter `swap` in das Verzeichnis `serie04`..

Aufgabe 4.7. Schreiben Sie eine Funktion `findmax`, die das Maximum eines gegebenen Vektors $v \in \mathbb{R}^n$ findet und sowohl den Wert des Maximums als auch den ersten Index, an dem dieser Wert auftritt, zurückgibt. Der Vektor soll, wie zuvor, von der Tastatur eingelesen werden.Speichern Sie den Source-Code unter `findmax.c` in das Verzeichnis `serie04`.

Aufgabe 4.8. 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 Funktion `mysin`, die für gegebene $x \in \mathbb{R}$ und $\varepsilon > 0$ den Wert $S_n(x)$ zurückliefert, sobald

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

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 `mysin.c` in das Verzeichnis `serie04`.

Aufgabe 4.9. Was ist die größte, was die kleinste Zahl, die von einer Variable des Typs `long` gespeichert werden kann? Schreiben Sie ein Programm, welches Die Bit-Größe von `long` berechnet und die beiden Zahlen ausgibt. Erzeugen Sie die gleiche Ausgabe schließlich für den Datentyp `unsigned int`. Speichern Sie den Source-Code unter `big.c` in das Verzeichnis `serie04`.

Aufgabe 4.10. Schreiben Sie eine Funktion `findme` die folgendes realisiert: Zu Beginn wird (z.B. durch eine andere Person) im Hauptprogramm eine beliebige Zahl $n \in \mathbb{N}$ von der Tastatur eingelesen und an `findme` übergeben. Ziel ist es nun diese Zahl zu erraten. Hierzu soll die Funktion den Benutzer so lange zu neuen Eingaben auffordern, bis die entsprechende Zahl getroffen ist. Die ursprüngliche Eingabe sollte natürlich wieder gelöscht werden. Damit das nicht allzu lange dauert soll die Funktion nach jeder Eingabe einen Tip in der Form

`Die gesuchte Zahl ist größer als die letzte Eingabe`,

bzw.

`Die gesuchte Zahl ist kleiner als die letzte Eingabe`

abgeben. Lassen Sie nun Ihren Tutor raten! Speichern Sie den Source-Code unter `findme.c` in das Verzeichnis `serie04`.