Dirk Praetorius,
Gregor Gantner

Wintersemester 2018/19
25.10.2018

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 2

Aufgabe 2.1. Schreiben Sie eine Funktion `bogenmass`, die einen im Gradmaß gegebenen Winkel $\theta \in \mathbb{R}^+$ ins Bogenmaß umrechnet. Dabei soll der Rückgabewert ψ in reduzierter Form als $\psi \in [0, 2\pi)$ zurückgegeben werden. Speichern Sie den Source-Code unter `bogenmass.c` in das Verzeichnis `serie02`.

Aufgabe 2.2. Schreiben Sie eine `void`-Funktion `kurvendiskussion`, die für eine quadratische Funktion $p(x) = a + bx + cx^2$ mit Koeffizienten $a, b, c \in \mathbb{R}$ eine Kurvendiskussion durchführt. Wenn vorhanden, berechne man das Extremum (und Art) und die Nullstellen. Anderenfalls gebe man aus, dass die Funktion kein Extremum bzw. keine Nullstelle besitzt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, das die Parameter a, b, c einliest und die Funktion aufruft. Speichern Sie den Source-Code unter `kurvendiskussion.c` in das Verzeichnis `serie02`.

Aufgabe 2.3. Schreiben Sie eine Funktion `punkte`, die überprüft, ob drei gegebene Punkte (x, y), (u, v) und (a, b) in $\mathbb{R}^2$ auf einer Geraden liegen. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem die sechs Parameter x, y, u, v, a, b eingelesen und das Resultat ausgegeben werden. Speichern Sie den Source-Code unter `punkte.c` in das Verzeichnis `serie02`.

Aufgabe 2.4. Schreiben Sie eine `void`-Funktion `quadrant`, die für einen Punkt $(x, y) \in \mathbb{R}^2$ ausgibt, ob (x, y) auf einer der Achsen des Koordinatensystems liegt. Falls nicht, soll ausgegeben werden, in welchem Quadranten (x, y) liegt. Schreiben Sie ferner ein Hauptprogramm, in dem $x, y \in \mathbb{R}$ eingelesen werden. Speichern Sie den Source-Code unter `quadrant.c` in das Verzeichnis `serie02`.

Aufgabe 2.5. Schreiben Sie eine rekursive Funktion `binomial`, die für zwei gegebene Ganzzahlen $0 \le k \le n$ den Binomialkoeffizienten $\binom{n}{k}$ berechnet und zurückgibt. Verwenden Sie dazu das Additionstheorem

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \quad \text{für } 1 \le k < n$$

mit $\binom{n}{0} = 1 = \binom{n}{n}$. Schreiben Sie ein aufrufendes Hauptprogramm, in dem die Ganzzahlen k und n eingelesen und das Ergebnis $\binom{n}{k}$ ausgegeben werden. Speichern Sie den Source-Code unter `binomial.c` in das Verzeichnis `serie02`.

Aufgabe 2.6. Schreiben Sie eine rekursive Funktion `division`, die für zwei gegebene Ganzzahlen $m \ge 0$ und $n > 0$ die Integer-Division m/n (Division ohne Rest) berechnet und zurückgibt. Die Funktion darf nur die arithmetischen Operationen `+` und `-` verwenden. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem m und n eingelesen werden und das Ergebnis m/n ausgegeben wird. *Hinweis:* Es gilt $x/y = 1 + (x - y)/y$ für $y \neq 0$. Speichern Sie den Source-Code unter `division.c` in das Verzeichnis `serie02`.

Aufgabe 2.7. 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 `fibonacci`, die zu gegebenem Index n das Folgenglied x_n berechnet und zurückgibt. Schreiben Sie ferner ein aufrufendes Hauptprogramm, in dem n über die Tastatur eingelesen und das Ergebnis ausgegeben werden. Speichern Sie den Source-Code unter `fibonacci.c` in das Verzeichnis `serie02`.

Aufgabe 2.8. Nach einer alten Legende gibt es in Hanoi einen Tempel, in dem sich folgende rituelle Anordnung befindet: Es handelt sich um 3 Pfosten und um 64 goldene Scheiben, die in der Mitte ein Loch haben, so dass sie auf die Pfosten gestapelt werden können. Die 64 Scheiben haben paarweise verschiedenen Durchmesser. Als der Tempel erbaut wurde, wurden diese Scheiben der Größe nach aufsteigend auf den ersten Pfosten gestapelt. Die Aufgabe der Priester in diesem Tempel besteht darin, diese Scheiben systematisch unter Zuhilfenahme des zweiten Pfostens so umzustapeln, dass sich diese am Schluss in derselben Anordnung wie zu Beginn auf dem dritten Pfosten befinden. Dabei sind folgende Regeln zu beachten:

- Die Scheiben dürfen nur einzeln verschoben werden.
- In jedem Schritt wird die oberste Scheibe eines Stapels auf einen der anderen Stapel gesetzt. D.h. es kann jeweils nur die erste Scheibe bewegt werden.
- Eine Scheibe darf nie auf einer anderen Scheibe kleineren Durchmessers zu liegen kommen.

Das Ende der Welt, so sagt die Legende, ist durch denjenigen Zeitpunkt vorherbestimmt, an dem die Priester diese Aufgabe erfüllt haben werden.
Sei n die Anzahl der Scheiben ($n = 64$ in der Legende). Ein rekursiver Algorithmus, der dieses Problem löst, lässt sich wie folgt formulieren: Um die obersten $m \leq n$ auf Pfosten i befindlichen Scheiben auf Pfosten j zu verschieben, verschiebt man

1. die obersten $m-1$ Scheiben von Pfosten i auf Pfosten $k \notin \{i, j\}$,
2. die grösste der besagten m Scheiben von Pfosten i auf Pfosten j,
3. und schliesslich die $m-1$ in Schritt 1 auf Pfosten k verschobenen Scheiben auf Pfosten j.

Die Wahl $m = n$, $i = 1$ und $j = 3$ löst das Problem. Schreiben Sie eine rekursive Funktion `void hanoi(int m, int i, int j)` die diesen Algorithmus implementiert. Jeder einzelne Schritt soll dabei am Display protokolliert werden. Zum Beispiel:

```
Eine Scheibe wandert vom 2. zum 3. Pfosten.
```

Schreiben Sie auch ein Hauptprogramm, das n über die Tastatur einliest und die Liste der Schritte ausgibt. Zum Testen verwende man $n \ll 64$, z.B. $n = 3, 4, 5$. Speichern Sie den Source-Code unter `hanoi.c` in das Verzeichnis `serie02`.