Winfried Auzinger
Klaus Felsenstein
Dirk Praetorius

SS 2011
(3. Mai 2011)

Übungsaufgaben zur VL Computermathematik

Serie 8

Aufgabe 8.1*. Speichern Sie die Ziffern eines Sudoku in einem ein 9×9 - `Array S`, z.B.:

9	7	6	4	2	1	3	8	5
8	3	4	6	9	5	1	2	7
1	2	5	7	8	3	4	9	6
7	4	9	1	3	2	5	6	8
3	6	2	5	4	8	7	1	9
5	1	8	9	6	7	2	3	4
2	9	7	8	1	4	6	5	3
4	8	3	2	5	6	9	7	1
6	5	1	3	7	9	8	4	2

Erstellen Sie eine Prozedur `check_sudoku(S)`, die folgendes überprüft:

- Ist das Sudoku syntaktisch korrekt gebaut, d.h. handelt es sich um ein `Array[1..9,1..9]` und ist in jedem Kästchen eine Zahl $\in \{0,\ldots,9\}$ eingetragen? (0 soll bedeuten, dass das betreffende Feld noch leer ist.)
- Ist das Sudoku konsistent, d.h. korrekt aufgesetzt (mit bzw. ohne Berücksichtigung von Farben)? Dabei ist Unvollständigkeit zugelassen, d.h. leere Felder (0) sind erlaubt.
- Ist das Sudoku sogar vollständig korrekt gelöst (mit bzw. ohne Berücksichtigung von Farben)?

Rückgabewert (z.B.):

–2 (nicht syntaktisch korrekt),
–1 (syntaktisch in Ordnung, aber nicht korrekt aufgesetzt/gelöst, d.h. inkonsistent),
0 (leer – lauter Nullen),
1 (unvollständig, korrekt aufgesetzt, aber nur ohne Berücksichtigung der Farben),
2 (unvollständig, korrekt aufgesetzt auch mit Berücksichtigung der Farben),
3 (vollständig, korrekt gelöst, aber nur ohne Berücksichtigung der Farben),
4 (vollständig, korrekt gelöst, auch mit Berücksichtigung der Farben).

Anmerkung: Versuchen Sie explizite Schleifen so weit wie möglich zu vermeiden. Praktisch ist die Selektion von Elementen mittels impliziter Schleifenkonstrukte (vgl. `?seq`).

Aufgabe 8.2*. Eine Abbildung $f\colon M \to \mathbb{Z}$ (M eine endliche Teilmenge von $\mathbb{Z}$) sei ein Form einer `listlist` gespeichert, z.B.

```
[[1,2],[2,3],[3,1]]      →   M={1,2,3},  f(1)=2, f(2)=3, f(3)=1
```

Erstellen Sie eine Prozedur `check_f(l)`, die folgendes überprüft:

- Ist `l` eine `listlist`? (das ist ein eigener Datentyp)
- Sind alle Einträge vom Typ `integer`?
- Ist durch `l` eine Abbildung f korrekt definiert?
- Ist diese Abbildung injektiv?

Rückgabewert (z.B.):

–2 (keine `listlist`),

–1 (nicht alles ganzzahlig),

0 (definiert keine Abbildung),

1 (definiert Abbildung, aber nicht injektiv);

bzw. falls die Abbildung injektiv ist, ist die Umkehrabbildung in Form einer analog gebauten `listlist` zurückzugeben.

Anmerkung: Die Reihenfolge (Sortierung) der Werte soll dabei egal sein. Überlegen Sie, wie Sie Sie die Abbildungs- bzw. Injektivitätseigenschaft effizient überprüfen (ohne Verwendung von Schleifen).

Aufgabe 8.3*. Produzieren Sie einen `pointplot` für den Graphen einer gemäß der vorhergehenden Aufgabe definierten Funktion (`?plots[pointplot]`). Interpretieren Sie die Aufgabenstellung dahingehend, dass der plot optisch möglichst gut aussehen soll. (`?plot/options, ?plot/color`). Dazu kann es erforderlich sein, mehrere Varianten des plots zu erzeugen und mittels `plots[display]` gemeinsam darzustellen:

```
with(plots);
...
p1 := pointplot(...):
p2 := pointplot(...):
...
display([p1,p2,...]);
```

Aufgabe 8.4*. Der *Euklid'sche Algorithmus* dient zur Berechnung des größten gemeinsamen Teilers $\mathrm{ggT}(m,n) = \mathrm{ggT}(n,m)$ zweier natürlicher Zahlen m und n. Pseudocode dazu:

$$\text{while } m \bmod n \neq 0 \;\; \{\, m,\, n := n,\, m \bmod n \,\}$$

und der Wert von m am Ende ist genau das gesuchte der gesuchte ggT.

Schreiben Sie eine Prozedur `ggTkgV(m,n)`, die darauf basierend den ggT und das kgV (kleinstes gemeinsames Vielfaches) zweier Zahlen $m, n \in \mathbb{N}$ zurückgibt. Testen Sie mit Hilfe der analogen Maple-Funktionen `gcd` und `lcm`.

Anmerkung:

– In einer Maple-Zuweisung der Form `a,b,... := c,d,...` wird zunächst auf der rechten Seite alles ausgewertet, und danach erst werden diese Werte den Variablen links zugewiesen. In diesem Sinn ist die doppelte Zuweisung in obigem Pseudocode zu verstehen.

– In dem Pseudocode werden keine Hilfsvariablen verwendet, sondern die Werte von n und m werden überschrieben. Dies funktioniert jedoch nicht innerhalb einer Maple-Prozedur (Argumente kann man nicht als lokale Variablen verwenden - probieren Sie es aus); d.h., man benötigt zwei lokale Hilfsvariablen.

– Schreiben Sie Ihre Prozedur so, dass jede erforderliche mod - Berechnung nur ein einziges mal erfolgt (in dem Pseudocode tritt die mod-Auswertung zweimal auf). Bleiben Sie jedoch bei der Variante mit `while` und verwenden Sie dafür *keine* weitere Hilfsvariable (also insgesamt nur zwei).

Anmerkung: Aufruf mit (m,n), $m < n$, führt im allgemeinen zu einer um einen Durchlauf kürzeren Schleife im Vergleich zu (n,m). Sie können das auch verfolgen, indem Sie Zwischenergebnisse innerhalb der Schleife mit `print(...)` ausgeben (generell nützlich zum Verfolgen eines Algorithmus bzw. zur Suche von Fehlern).

Aufgabe 8.5. Für ein $n \in \mathbb{N}$ seien n verschiedene reelle Zahlen $\xi_0, \ldots, \xi_n$ gegeben. Das k - te *Lagrange-Polynom* zu den ξ_j ist definiert durch

$$L_k(x) := \prod_{\substack{\ell=0 \\ \ell \neq k}}^{n} \frac{x - \xi_\ell}{\xi_k - \xi_\ell} \,.$$

Dann gilt laut Konstruktion $L_k(\xi_j) = \delta_{k,j}$.

Implementieren Sie diese in Form einer Prozedur `lagrange_polynom(x,xi,k)`, wobei die ξ_k in einem `Array xi` (Index beginnend mit 0) übergeben werden. Verwenden Sie dies, um zu gegebenen Funktionswerten $f_0, \ldots, f_n$ (zu

den Stellen $x_0, \ldots, x_n$, analog in einem `Array f` vorgegeben) das Interpolationspolynom $p(x)$ vom Grad n in der Form

$$p(x) = \sum_{k=0}^{n} f_k \, L_k(x)$$

zu generieren. (Dies ist das eindeutige Polynom vom Grad $\leq n$ mit $p(\xi_k) = f_k$, $k = 0 \ldots n$.)

Konkret: Implementieren Sie die Auswertung von $p(x)$ an einer Stelle x in Form einer Prozedur `p := proc(x,xi,f)`.

Aufgabe 8.6. Gegeben seien zwei differenzierbare Funktionen $x(t)$ und $y(t)$, die eine Kurve in der Ebene beschreiben: Die Punkte auf der Kurve sind gegeben durch die Parameterdarstellung

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}, \quad a \leq t \leq b,$$

und zwar mittels einer Prozedur `P := proc(t)`, die den betreffenden Kurvenpunkt (in einer Liste der Länge 2) zurückliefert. (Die Koordinaten $x(t)$ und $y(t)$ werden dabei intern innerhalb von `P` definiert.)

Erstellen Sie weiters eine Prozedur `bogenlaenge(P,a,b)`, die (mit Hilfe von `diff` bzw. `D` sowie `int`) die Bogenlänge einer so definierten Kurve zwischen den Punkten $P(a)$ und $P(b)$ berechnet. Die Formel für die Bogenlänge lautet

$$\int_a^b |P'(s)| \, ds,$$

wobei $|\cdot|$ die euklidische Vektorlänge ist, und

$$P'(t) = \begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix}$$

der Tangentialvektor an die Kurve an der Stelle $P(t)$.

Ein sehr einfaches Beispiel: Umfang eines Kreises (Darstellung in Polarkoordinaten mit t = Winkel).

Aufgabe 8.7. Erstellen Sie zwei Prozeduren `innerprod3(a,b)` und `outerprod3(a,b)`, die zwei Listen `a` und `b` der Länge 3 als Vektoren im $\mathbb{R}^3$ interpretieren und das innere (Skalarprodukt, $a \cdot b$) bzw. das äußere (Vektorprodukt, $a \times b$, letzteres wieder als Liste) zurückgeben. Falls `a` oder `b` nicht die Länge 3 hat, ist jeweils `NULL` zurückzugeben.

'Beweisen' Sie damit die Tatsache $a \times b \perp a$, $a \times b \perp b$, und die Identitäten

$$|a \times b|^2 = |a|^2 \, |b|^2 - (a \cdot b)^2 \quad \text{(Formel von Lagrange)},$$
$$a \times (b \times c) = (a \cdot c)\, b - (a \cdot b)\, c,$$
$$(a \times b) \cdot (c \times d) = (a \cdot c)(b \cdot d) - (b \cdot c)(a \cdot d),$$

indem Sie beliebige 'generische' Vektoren (ohne konkrete Werte) übergeben.

Aufgabe 8.8. Erstellen Sie eine Prozedur `psearch(m::posint,n::posint)` die zu zwei `posint`-Zahlen `m` und `n` festellt an welcher Stelle die (komplette) Dezimalziffernfolge von `m` in der Dezimalziffernfolge von `n` zum ersten mal auftritt (ansonsten `NULL`).

Beispiel: `m:=1230; n:=10213230123010320;` die Antwort lautet 8.

Anmerkung: Basteln Sie einen Algorithmus, der nur arithmetische Operationen verwendet.

Vorschlag: Falls `m` bereits an der ersten Position von `n` auftritt, bedeutet dies: Extrahiert man die ersten μ Ziffern aus `n`, wobei μ = Anzahl der Ziffern in `m`, und ergibt dies Übereinstimmung mit `m`, so ist das Problem gelöst (Antwort = 1). Ansonsten geht man Stelle für Stelle innerhalb von `n` weiter. Man benötigt also entsprechende Hilfsprozeduren. Beachten Sie die Definition von `ilog10`, und verwenden Sie `iquo` und `irem` für ganzzahlige Division mit Rest. (Das Ganze ließe sich auch ein wenig anders organisieren.)

Beachten Sie, dass Nullen als Ziffern innerhalb (oder am Ende von) `m` und `n` auftreten können. Damit muss man richtig umgehen. In einer ersten Version könnte man auch davon ausgehen, dass Nullen nicht zugelassen sind (etwas einfacher).