Dirk Praetorius,
Michele Ruggeri

Sommersemester 2017
09.05.2017

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 8

Aufgabe 8.1. Schreiben Sie eine rekursive Funktion `papierschnitt`, die alle Möglichkeiten visualisiert, wie ein Papierbogen der ganzzahligen Länge n in Papierbahnen der Länge 1 und 2 geschnitten werden kann. D.h. man stelle eine natürliche Zahl n auf alle möglichen Weisen als Summe $n = \sum_{j=1}^{k} \sigma_j$ mit Summanden $\sigma_j \in \{1, 2\}$ dar. Dabei soll die Reihenfolge beachtet werden. Für $n = 4$ gibt es beispielsweise 5 Möglichkeiten:

- $4 = 2 + 2$
- $4 = 2 + 1 + 1$
- $4 = 1 + 2 + 1$
- $4 = 1 + 1 + 2$
- $4 = 1 + 1 + 1 + 1$

Schreiben Sie ein aufrufendes Hauptprogramm, in dem n eingelesen wird und alle Möglichkeiten ausgegeben werden. Speichern Sie den Source-Code unter `papierschnitt.m` in das Verzeichnis `serie08`.

Aufgabe 8.2. Schreiben Sie einen Strukturdatentyp `Date` zur Speicherung alle Daten ab 01.01.1900. Die Struktur besteht aus drei `int`-Members (Tag, Monat und Jahr). Schreiben Sie die Funktionen

- `Date* newDate(int d, int m, int y)`,
- `Date* delDate(Date* date)`,

sowie die sechs Zugriffsfunktionen

- `void setDateDay(Date* date, int d)`,
- `void setDateMonth(Date* date, int m)`,
- `void setDateYear(Date* date, int y)`,
- `int getDateDay(Date* date)`,
- `int getDateMonth(Date* date)`,
- `int getDateYear(Date* date)`.

Schreiben Sie die Funktion `int isMeaningful(Date* date)`, die die Zulässigkeit einer Datum überprüft (Rückgabewert `1` bei Zulässigkeit, sonst `0`). Die Datum 31.02.2013 z.B. ist nicht zulässig (Schaltjahre nicht vergessen!). Schreiben Sie ferner ein aufrufendes Hauptprogramm, um Ihre Implementierung entsprechend zu testen. Speichern Sie den Source-Code, aufgeteilt in Header-Datei `datum.h` und `datum.c`, in das Verzeichnis `serie08`.

Aufgabe 8.3. Schreiben Sie einen Strukturdatentyp `Person` zur Speicherung von personenbezogenen Daten. Die Struktur besteht aus vier Members: `firstname` (`char*`), `surname` (`char*`), `address` (`Address*`) und `birthday` (`Date*`). Schreiben Sie außerdem alle nötigen Funktionen um mit dieser Struktur arbeiten zu können. Verwenden Sie die Struktur `Date` aus Aufgabe 8.2 sowie die Struktur `Address` aus der Vorlesung (Folie 183). Schreiben Sie die Funktion `Person* whoIsOlder(Person* a, Person* b)`, die die Alter zwei Personen vergleicht und die jüngere Person zurückgibt. Testen Sie Ihre Implementierung entsprechend! Speichern Sie den Source-Code, aufgeteilt in Header-Datei `person.h` und `person.c`, in das Verzeichnis `serie08`.

Aufgabe 8.4. Schreiben Sie einen Strukturdatentyp `SquareMatrix` zur Speicherungen quadratischer Matrizen $A \in \mathbb{R}^{n\times n}$. Hierbei sollen die Einträge der Matrix spaltenweise als `double*`, sowie die Größe $n \in \mathbb{N}$ abgespeichert werden. Im Gegensatz zur üblichen Indizierung in C (bzw. zur Indizierung in der Vorlegung) soll die Indizierung der Matrixeinträge a_{jk} in Ihrer Struktur `SquareMatrix` von $j, k = 1$ bis n laufen (wie in der Mathematik üblich). Schreiben Sie außerdem alle nötigen Funktionen um mit dieser Struktur arbeiten zu können, d.h. implementieren Sie `newSquareMatrix`, `delSquareMatrix`, `getSquareMatrixN`, `getSquareMatrixEntry` und `setSquareMatrixEntry`. Testen Sie Ihre Implementierung entsprechend! Speichern Sie den Source-Code, aufgeteilt in Header-Datei `squarematrix.h` und `squarematrix.c`, in das Verzeichnis `serie08`.

Aufgabe 8.5. Die Frobeniusnorm einer quadratischen Matrix $A \in \mathbb{R}^{n\times n}$ ist durch

$$\|A\|_F := \left(\sum_{j,k=1}^{n} a_{jk}^2 \right)^{1/2}$$

definiert. Schreiben Sie eine Funktion `double frobeniusnorm(SquareMatrix* A)`, die die Frobeniusnorm einer quadratischen Matrix $A \in \mathbb{R}^{n\times n}$ berechnet und zurückgibt. Verwenden Sie die Struktur `SquareMatrix` aus Ausgabe 8.4. Testen Sie Ihre Implementierung entsprechend! Speichern Sie den Source-Code unter `frobeniusnorm.c` in das Verzeichnis `serie08`.

Aufgabe 8.6. Der Laplacesche Entwicklungssatz für Determinanten besagt, dass für ein beliebiges $j \in \{1, \ldots, n\}$ gilt, dass

$$\det A = \sum_{k=1}^{n} (-1)^{j+k} \cdot a_{jk} \cdot \det A_{jk},$$

wobei A_{jk} die $(n-1)\times(n-1)$-Untermatrix von A ist, die durch Streichen der j-ten Zeile und k-ten Spalte entsteht. Schreiben Sie eine rekursive Funktion `double detlaplace(SquareMatrix* A)`, die die Determinante $\det(A)$ einer Matrix $A \in \mathbb{R}^{n\times n}$ mit Hilfe des Laplaceschen Entwicklungssatzes berechnet und zurückgibt. Verwenden Sie die Struktur `SquareMatrix` aus Ausgabe 8.4. Testen Sie Ihre Implementierung entsprechend! Speichern Sie den Source-Code unter `detlaplace.c` in das Verzeichnis `serie08`.

Aufgabe 8.7. Nicht jede Matrix $A \in \mathbb{R}^{n\times n}$ hat eine normalisierte LU-Zerlegung $A = LU$, d.h.

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ \ell_{21} & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \ell_{n1} & \ldots & \ell_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1n} \\ 0 & u_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & u_{n-1,n} \\ 0 & \ldots & 0 & u_{nn} \end{pmatrix}.$$

Wenn aber A eine normalisierte LU-Zerlegung besitzt, so gilt

$$\begin{aligned} u_{ik} &= a_{ik} - \sum_{j=1}^{i-1} \ell_{ij} u_{jk} \quad \text{für } i = 1, \ldots, n, \quad k = i, \ldots, n, \\ \ell_{ki} &= \frac{1}{u_{ii}} \left(a_{ki} - \sum_{j=1}^{i-1} \ell_{kj} u_{ji} \right) \quad \text{für } i = 1, \ldots, n, \quad k = i+1, \ldots, n, \\ \ell_{ii} &= 1 \quad \text{für } i = 1, \ldots, n, \end{aligned}$$

wie man leicht über die Formel für die Matrix-Matrix-Multiplikation zeigen kann. Alle übrigen Einträge von $L, U \in \mathbb{R}^{n\times n}$ sind Null. Schreiben Sie eine Funktion `SquareMatrix* computeLU(SquareMatrix* A)`, die die LU-Zerlegung von A berechnet und zurückgibt. Dazu überlege man, in welcher Reihenfolge man die Einträge von L und U berechnen muss, damit die angegebenen Formeln wohldefiniert sind (d.h. alles was benötigt wird, ist bereits zuvor berechnet worden). Verwenden Sie die Struktur `SquareMatrix` aus Ausgabe 8.4. Schreiben Sie ein aufrufendes Hauptprogramm, in dem Sie die Funktion `computeLU` an einen geeigneten Beispiel testen. Speichern Sie den Source-Code unter `computeLU.c` in das Verzeichnis `serie08`.

Aufgabe 8.8. Um die Determinante einer Matrix $A \in \mathbb{R}^{n \times n}$ zu berechnen, ist der Laplacesche Entwicklungssatz aus Aufgabe 8.6 kein geeignetes Mittel. Es ist besser, man berechnet die normalisierte LU-Zerlegung aus Aufgabe 8.7. Es gilt nämlich $\det(A) = \det(L)\det(U) = \det(U) = \prod_{j=1}^{n} u_{jj}$. Schreiben Sie eine Funktion `double detLU(SquareMatrix* A)`, die die Determinante einer Matrix A mittels der normalisierten LU-Zerlegung berechnet und zurückgibt. Verwenden Sie die Struktur `SquareMatrix` aus Ausgabe 8.4. Testen Sie Ihre Implementierung entsprechend! Speichern Sie den Source-Code unter `detLU.c` in das Verzeichnis `serie08`.