Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
13.01-15.01.2016

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 11

Aufgabe 11.1. Schreiben Sie eine Klasse `Alkohol` zur Speicherung von verschiedenen alkoholischen Getränken. Diese soll folgende Member-Variablen enthalten: Name, Alkoholgehalt in Prozent, Preis in €. Weiters soll für diese Klasse neben einem passenden Konstruktor auch der `operator<` definiert werden. Dieser soll nach besserem Verhältnis $\frac{\text{Vol.\%}}{€}$ bewerten. Definieren Sie auch die Methoden `getName()`, `getPreis()` und `getVolProz()`.
Hinweis: Für die Überladung des Operators `<` beachte man die allgemeine Syntax

```
bool operator<(const type& lhs, const type& rhs);
```

Hier bezeichnet `type` einen beliebigen Datentyp. In unserem Fall ist das also `Alkohol`.

Aufgabe 11.2. Schreiben Sie eine Klasse `Matrix` zur Speicherung von Matrizen der Größe $m \times n$. Die Einträge sollen hierbei als langer `double*`-Vektor der Länge mn gespeichert werden. Schreiben Sie Methoden, welche die Möglichkeit bieten Einträge zu erstellen, bzw. auszulesen. Achten Sie hierbei auf eventuelle Sicherheitsabfragen. Schreiben Sie einen Konstruktor, der bei übergebenen Werten $m, n \in \mathbb{N}$ eine Nullmatrix der Größe $m \times n$ erzeugt, sowie einen Standardkonstruktor, der eine leere Matrix der Größe 0×0 generiert. Natürlich darf auch ein entsprechender Destruktor nicht fehlen. Speichern Sie den Source-Code unter `matrix.{hpp/cpp}` in das Verzeichnis `serie11`.

Aufgabe 11.3. Erweitern Sie die Klasse `Matrix` aus Aufgabe 11.2 mit

- einen Zuweisungsoperator,
- einen Kopierkonstruktor,
- einen Operator um sinnvoll $-A$ ausführen zu können,
- einer Methode `scanMatrix(int n,int m)`, die eine $n \times m$-Matrix von der Tastatur einliest,
- einer Methode `printMatrix()`, die die Matrix am Bildschirm ausgibt,
- und einer Methode `transpose` zum Überschreiben einer gespeicherten Matrix mit der transponieren Matrix. (Hinweis: Die Transponierte $A^T \in A \in \mathbb{R}^{n\times m}$ einer Matrix $A \in \mathbb{R}^{m\times n}$ ist durch die Beziehung $\left(A^T\right)_{jk} = A_{kj}$ definiert. Es gilt dann $\left(A^T\right)^T = A$.)

Schreiben Sie weiters ein aufrufendes main Programm um ihre implementierten Methoden zu testen. Speichern Sie den Source-Code unter `matrix.{hpp/cpp}` in das Verzeichnis `serie11`.

Aufgabe 11.4. Eine untere Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ mit

$$L = \begin{pmatrix} \ell_{11} & & & & \mathbf{0} \\ \ell_{21} & \ell_{22} & & & \\ \ell_{31} & \ell_{32} & \ell_{33} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \ell_{n1} & \ell_{n2} & \ell_{n3} & \dots & \ell_{nn} \end{pmatrix}$$

hat höchstens $\frac{n(n+1)}{2} = \sum_{j=1}^{n} j$ nicht-triviale Einträge. Schreiben Sie eine Klasse `matrixL`, in der neben der Dimension $n \in \mathbb{N}$ die Koeffizienten L_{ij} in einem dynamischen Vektor der Länge $\frac{n(n+1)}{2}$ gespeichert werden. Speichern Sie L zeilenweise. Implementieren Sie die folgenden Funktionalitäten:

- Konstruktor, Copy-Konstruktor, Destruktor,
- Zuweisungssoperator,
- Zugriff auf die Koeffizienten mittels `L(i,j)` und
- die Möglichkeit eine untere Dreiecksmatrix `L` mit `cout << L` auszugeben.

Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierung testen.

Aufgabe 11.5. Überladen Sie den Operator `+` für die Klasse `MatrixL` aus Aufgabe 11.4 um zwei untere Dreiecksmatrizen bei passenden Dimensionen addieren zu können. Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierung testen.

Aufgabe 11.6. Beweisen Sie mit der Formel des Matrix-Matrix-Produktes, dass das Produkt zweier unterer Dreiecksmatrizen eine untere Dreiecksmatrix ist. Überladen Sie den Operator `*` für die Klasse `MatrixL` aus Aufgabe 11.4 sodass Sie das Matrixprodukt für zwei untere Dreiecksmatrizen bei passenden Dimensionen berechnen können. Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierungen testen.

Aufgabe 11.7. Gegeben sei eine untere Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ mit $\ell_{jj} \neq 0$ für alle $j = 1, \ldots, n$. Zu gegebenem $b \in \mathbb{R}^n$ existiert dann ein eindeutiges $x \in \mathbb{R}^n$ mit $Lx = b$. Implementieren Sie die Möglichkeit, für eine untere Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ und einen Vektor $b \in \mathbb{R}^n$ das System $Lx = b$ mittels `x=L|b` zu lösen. L ist dabei vom Typ `Matrix` aus Aufgabe 11.4 und b ist dabei vom bekannten Typ `Vector` aus der Vorlesung. Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierung testen.

Aufgabe 11.8. Erklären Sie den Unterschied zwischen Referenzen und Pointern mit eigenen Worten. Schreiben Sie exemplarisch einen Code, welcher die Inhalte zweier Variablen vertauscht, einmal mit Pointern und einmal mit Referenzen. Welche Vorteile bringt die Verwendung von Referenzen gegenüber Pointern mit sich? Welche Nachteile?