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. Write a class `Alcohol` for the storage of different alcoholic drinks. The class should contain the following members: name, alcoholic strength percent, price in €. Moreover, implement an appropriate constructor and overload `operator<`, that compares two objects of the class with respect to the ratio $\frac{\text{Vol.\%}}{\text{€}}$. Additionally, implement the methods `getName()`, `getPrice()`, and `getVolPercent()`.
Hint: In general, the operator `<` is overloaded by the syntax

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

Here, `type` is an arbitrary datatype. In our case it is `Alcohol`.

Aufgabe 11.2. Write a class `Matrix` for the storage of $m \times n$ matrices $A \in \mathbb{R}^{m\times n}$. The entries should be stored columnwise by a `double*`-array of length mn. Write `get/set`-methods for the entries of the matrix and `get`-methods for the dimensions. Moreover, write a constructor with input $m, n \in \mathbb{N}_0$, that allocates memory for a $m\times n$ matrix and initializes all entries with 0. Implement the standard constructor which generates a 0×0-matrix as well as a destructor which frees allocated memory. Save your source code as `matrix.`{`hpp/cpp`} into the directory `serie11`.

Aufgabe 11.3. Extend the class `Matrix` from Exercise 11.2 by the following methods:

- an assignment operator,
- a copy-constructor,
- an operator to be able to perform $-A$,
- a method `scanMatrix(int n,int m)`, which reads a $n \times m$-Matrix from the keyboard,
- a method `printMatrix()` which prints a matrix on screen,
- and a method `transpose`, which overwrites a stored matrix with the transposed matrix. (Hint: The transposed $A^T \in A \in \mathbb{R}^{n\times m}$ of a matrix $A \in \mathbb{R}^{m\times n}$ is defined by the condition $\left(A^T\right)_j k = A_k j$. We then have $\left(A^T\right)^T = A$.)

Moreover, write a main programm to test your implemented methods. Save your source code as `matrix.`{`hpp/cpp`} into the directory `serie11`.

Aufgabe 11.4. A lower triangular matrix $L \in \mathbb{R}^{n\times n}$ with

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

has at most $\frac{n(n+1)}{2} = \sum_{j=1}^{n} j$ nontrivial coefficients. Write a class `matrixL` to save the coefficients L_{ij} in a dynamical vector with length $\frac{n(n+1)}{2}$ together with the dimension $n \in \mathbb{N}$. Save the matrix L row-wise. Implement the following features:

- constructor, copy-constructor, destructor,

- assignment operator,
- access to the coefficients via `L(i,j)` and
- the possibility to print a lower triangular matrix `L` on screen via `cout << L`.

Moreover, write a main-program to test your implementation.

Aufgabe 11.5. Overload the operator `+` for the class `MatrixL` from Exercise 11.4 to be able to add to lower triangular matrices with matching dimensions. Moreover, write a main-programm to test your implementation.

Aufgabe 11.6. Use the formula for the matrix-matrix product to show that the product of two lower triangular matrices is a lower triangular matrix. Then, overload the operator `*` for the class `MatrixL` from Exercise 11.4 to be able to perform the matrix-matrix product for two lower triangular matrices with matching dimensions. Moreover, write a main-program to test your implementation.

Aufgabe 11.7. Let $L \in \mathbb{R}^{n\times n}$ be a lower triangular matrix such that $\ell_{jj} \neq 0$ for all $1 \leq j \leq n$. Given $b \in \mathbb{R}^n$, there exists a unique $x \in \mathbb{R}^n$ such that $Lx = b$. Implement also the feature to solve the system $Lx = b$ for a lower triangular matrix $L \in \mathbb{R}^{n\times n}$ and a vector $b \in \mathbb{R}^n$ by using the command `x=L|b`. L has the type `MatrixL` from Exercise 11.4 and b has the well-known type `Vector` from the lecture. Moreover, write a main-program to test your implementation.

Aufgabe 11.8. Explain the differences between references and pointers. Write some code which swaps the values of two variables. Implement a version which uses pointers and then implement a second version which uses references. What are the advantages of using references? What are the disadvantages?