Dirk Praetorius
Michele Ruggeri

Sommersemester 2015
23.06.2015

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 13

Aufgabe 13.1. Implement a class `Person` which contains the members `name` and `address`. Derive a class `Student` from `Person`, that contains the additional data-fields `matriculationNumber` and `study`. Derive another class `Worker` that contains the additional data-fields `salary` and `work`. Write `set/get` functions, constructors and destructors for these. Moreover, write a main progam to test your implementation!

Aufgabe 13.2. Implement the method `virtual void print()` in the basis class `Person` from exercise 13.1. The method should print out the name and address of a person. Redefine this function in the derived classes `Student` and `Worker` (the additional data-fields should also be printed out). Moreover, write a main programm for testing the `print`-methods of the different classes and explain the usage of `virtual` in this context.

Aufgabe 13.3. Consider the class `Matrix` and the derived class `SquareMatrix` from the lecture. Implement the method `computeLU`, that computes the LU-factorization, for the class `SquareMatrix`. The return value (a matrix $R \in \mathbb{R}^{n\times n}$ is again of the type `SquareMatrix`, where the triangular matrices L and U should be stored in R. The diagonal of L does not need to be stored. Why?
Not every matrix $A \in \mathbb{R}^{n\times n}$ has a normalized LU-factorization $A = LU$, i.e.,

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

In the case there exists such a factorization, it holds

$$u_{ik} = a_{ik} - \sum_{j=1}^{i-1} \ell_{ij} u_{jk} \quad \text{for } i = 1, \dots, n, \quad k = i, \dots, n,$$

$$\ell_{ki} = \frac{1}{u_{ii}} \Big(a_{ki} - \sum_{j=1}^{i-1} \ell_{kj} u_{ji} \Big) \quad \text{for } i = 1, \dots, n, \quad k = i+1, \dots, n,$$

$$\ell_{ii} = 1 \quad \text{for } i = 1, \dots, n,$$

which can be verified by using the formula for the matrix-matrix multiplication.

Aufgabe 13.4. The determinant of a matrix $A \in \mathbb{R}^{n\times n}$ can be computed with the normalized LU-factorization from exercise 13.3. It holds $\det(A) = \det(L)\det(U) = \det(U) = \prod_{j=1}^{n} u_{jj}$. Extend the class `SquareMatrix` by the method `detLU`, that computes and returns the determinant. The matrix A should not be overwritten.

Aufgabe 13.5. Extend the class `SquareMatrix` by the method `solveLU`, that computes the solution of a system of linear equations $Ax = b$ as follows. First compute the LU-factorization $A = LU$, then solve first $Ly = b$ and afterwards $Ux = y$. Systems of equations containing triangular matrices can be solved analoguosly to exercise 12.8. Test your code on a suitable example.

Aufgabe 13.6. What is the computational cost of the LU-factorization from exercise 13.3? Write down your results in the $\mathcal{O}$-notation.

Aufgabe 13.7. What is the computational cost to solve a linear equation system via LU-factorizatoin from exercise 13.5? Write down your results in the $\mathcal{O}$-notation.

Aufgabe 13.8. Explain the differences between `public`-, `private`-, und `protected`-inheritance on the basis of a suitable exapmle.