Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
20.-21.01.2016

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 12

Aufgabe 12.1. Implementieren Sie eine Klasse `Person`, welche die Datenfelder `name` und `adresse` enthält. Leiten Sie von dieser Klasse eine Klasse `Student` ab, welche die zusätzlichen Datenfelder `matrikelnummer` und `studium` enthält. Leiten Sie von der Klasse `Person` auch eine Klasse `Arbeiter` ab. Erweitern Sie diese Klasse um die Datenfelder `gehalt` und `arbeit`. Schreiben Sie für alle Klassen die zugehörigen Zugriffsfunktionen, Konstruktoren und Destruktoren. Schreiben Sie ein main-Programm und testen Sie ihre Implemtierung! Erklären Sie in diesem Zusammenhang den Unterschied zwischen `public`-, `private`-, und `protected`-Vererbung.

Aufgabe 12.2. Erstellen Sie für die Basisklasse `Person` aus Aufgabe 12.1 eine Methode `void print()`, welche den Namen und die Adresse einer Person am Bildschirm ausgibt. Redefinieren Sie diese Funktion dann jeweils für die Klassen `Student` und `Arbeiter` (Es sollen die zusätzlich definierten Datenelemente auch ausgegeben werden). Schreiben Sie dann noch ein Hauptprogramm, in welchem die `print`-Funktionen der verschiedenen Klassen getestet werden sollen. Erklären Sie in diesem Zusammenhang auch den Begirff `virtual` .

Aufgabe 12.3. Wir betrachten die Klasse `Matrix` und die davon abgeleitete Klasse `SquareMatrix` aus der VO. Implementieren Sie für die Klasse `SquareMatrix` die Methode `computeLU`, welche die LU-Zerlegung berechnet. Der Rückgabewert (Matrix $R \in \mathbb{R}^{n\times n}$) sei dabei wieder vom Type `SquareMatrix`, wobei die beiden Dreiecksmatrizen L und U in R gespeichert werden sollen. Die Diagonale von L muss hierbei nicht explizit gespeichert werden. (Warum?)
Nicht jede Matrix $A \in \mathbb{R}^{n\times n}$ hat eine normalisierte LU-Zerlegung $A = LU$, d.h.

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

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

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

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

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

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.

Aufgabe 12.4. Die Determinante einer Matrix $A \in R^{n\times n}$ kann über die normalisierte LU-Zerlegung aus Aufgabe 12.3 berechnet werden. Es gilt namlich $\det(A) = \det(L)\det(U) = \det(U) = \prod_{j=1}^{n} u_{jj}$. Erweitern Sie die Klasse `SquareMatrix` um eine Methode `detLU`, die die Determinante über die LU-Zerlegung berechnet und zurückgibt. Die Matrix selbst soll hierbei nicht überschrieben werden.

Aufgabe 12.5. Erweitern Sie die Klasse `SquareMatrix` um eine Methode `solveLU`, welche die Lösung eines Gleichungsystem der Form $Ax = b$ mit Hilfe der LU-Zerlegung folgendermaßen berechnet. Für die Matrix $A = LU$ löst man zuerst $Ly = b$ und anschließend $Ux = y$. Gleichungssysteme mit Dreiecksmatrizen können analog zu Aufgabe 11.8 gelöst werden. Testen Sie Ihren Code an einem geeigneten Beispiel.

Aufgabe 12.6. Welchen Aufwand besitzt die LU-Zerlegung aus Aufgabe 12.3? Schreiben Sie das Ergebnis in der $\mathcal{O}$-Notation auf und erklären Sie wie sie auf das Ergebnis gekommen sind.

Aufgabe 12.7. Welchen Aufwand hat das Lösen eines linearen Gleichungssystems aus Aufgabe 12.5? Schreiben Sie das Ergebnis in der $\mathcal{O}$-Notation auf und erklären Sie wie sie auf das Ergebnis gekommen sind.

Aufgabe 12.8. Wie lautet die Ausgabe des folgenden Programms? Erklären Sie warum!

```
#include <iostream>
using std::cout ;
using std::endl ;

class Basisklasse {
protected :
  int N;
public :
  Basisklasse( int n = 0) {
   N = n;
   cout << " Konstr . Basisklasse , N = " << N << endl;
  }
  virtual ~Basisklasse() {
   cout << " Destr . Basisklasse , N = " << N << endl;
  }
  virtual void print() {
   klasse();
   cout << " N = " << N << endl;
  }
  void Add() {
  }
  virtual void klasse() const {
   cout << " In Basisklasse aber virtual ,";
  }
  void klasse() {
   cout << " In Basisklasse ,";
  }
};


class Abgeleitet : public Basisklasse {
public :
  Abgeleitet( int n = 0) {
   N = n;
   cout << " Konstr . Abgleitet , N = " << N << endl;
  }
  ~Abgeleitet() {
   cout << " Destr . Abgeleitet , N = " << N << endl;
  }
  void print() const {
   klasse();
   cout << " const N =" << N << endl;
  }
  void print() {
   klasse();
   cout << " N = " << N << endl;
  }
  void Add(){
```

```
   N = N + 100;
  }
  void klasse() const {
   cout << " In Abgeleitet fuer const , ";
  }
  void klasse() {
   cout << " In Abgeleitet ,";
  }
};


int main() {
  Basisklasse dp(1);
  Abgeleitet mr(10);
  Basisklasse * bs = & mr;
  {
   const Abgeleitet ah(200);
   dp.Add();
   mr.Add();
   bs->Add();
   ah.print();
  }
  dp.print();
  mr.print();
  bs->print();
  return 0;
}
```