Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (3 Punkte):
Aufgabe 2 (1 Punkt):
Aufgabe 3 (2 Punkte):
Aufgabe 4 (4 Punkte):
Aufgabe 5 (2 Punkte):
Aufgabe 6 (2 Punkte):
Aufgabe 7 (4 Punkte):
Aufgabe 8 (4 Punkte):
Aufgabe 9 (4 Punkte):
Aufgabe 10 (2 Punkte):
Aufgabe 11 (2 Punkte):
Aufgabe 12 (5 Punkte):
Aufgabe 13 (5 Punkte):

Gesamtpunkte (40 Punkte):

Schriftlicher Test (120 Minuten)
VU Einführung ins Programmieren für TM

23. Januar 2017

Hinweis. In den folgenden Aufgaben betrachten wir Matrizen $A \in \mathbb{R}^{n\times n}$ als Objekte der C++ Klasse `Matrix`, die unten definiert ist. Neben Konstruktor, Kopierkonstruktor, Destruktor und Zuweisungsoperator, gibt es eine Methode, um die Dimension n auszulesen (`size`), die Zeilensummennorm zu berechnen (`norm`) und zu bestimmen, ob A eine untere Dreiecksmatrix ist. Die Koeffizienten A_{jk} erhält man mittels `A(j,k)`, wobei die Indizes $j, k = 1, \ldots, n$ im mathematisch üblichen Sinn verwendet werden. Intern wird eine Matrix $A \in \mathbb{R}^{n\times n}$ spaltenweise in einem dynamischen Vektor der Länge n^2 gespeichert. Ferner werden die Operatoren `+` und `*` mit der üblichen Bedeutung überladen.

```
class Matrix {
private:
  int n;
  double* coeff;

public:
  Matrix(int n=0, double init=0);
  Matrix(const Matrix&);
  ~Matrix();
  Matrix& operator=(const Matrix&);

  int size() const;
  double norm() const;
  int isLowerTriangular() const;

  const double& operator()(int, int) const;
  double& operator()(int, int);

};

const Matrix operator+(const Matrix&, const Matrix&);
const Matrix operator*(const Matrix&, const Matrix&);
```

Aufgabe 1 (3 Punkte). Schreiben Sie den Konstruktor der Klasse `Matrix`, der eine Matrix $A \in \mathbb{R}^{n\times n}$ anlegt, wobei der optionale Parameter `init` der Initialisierungswert für die Koeffizienten sei (d.h. $A_{jk} = \texttt{init}$ für alle $j, k = 1, \ldots, n$). Stellen Sie mittels `assert` sicher, dass die Dimension $n \geq 0$ ist. Für $n = 0$ werde die leere Matrix angelegt (d.h. es wird kein Speicher allokiert).

Lösung zu Aufgabe 1.

Aufgabe 2 (1 Punkt). Schreiben Sie den Destruktor der Klasse `Matrix`.

Lösung zu Aufgabe 2.

Aufgabe 3 (2 Punkte). Was ist die Bedeutung der Zeile

```
const Matrix& A = *this;
```

wenn Sie diese in einer Methode der Klasse `Matrix` verwenden? Erklären Sie die Syntax!

Aufgabe 4 (4 Punkte). Schreiben Sie den Zuweisungsoperator der Klasse `Matrix`.

Lösung zu Aufgabe 4.

Aufgabe 5 (2 Punkte). Die Matrix $A \in \mathbb{R}^{n \times n}$ wird intern spaltenweise als Vektor $a \in \mathbb{R}^{n^2}$ gespeichert. Beachten Sie, dass die Koeffizienten A_{jk} mit Indizes $j, k \in \{1, \dots, n\}$ indiziert werden, während die Koeffizienten a_ℓ des Speichervektors in C++ mit Indizes $\ell \in \{0, \dots, n^2 - 1\}$ indiziert werden. Leiten Sie eine Formel her, die für ein zulässiges Indexpaar (j, k) den zugehörigen Index ℓ liefert. Begründen Sie Ihre Formel!

Lösung zu Aufgabe 5.

Aufgabe 6 (2 Punkte). Schreiben Sie den Zugriffsoperator `( )` für nicht-konstante Objekte der Klasse `Matrix` für den Koeffizientenzugriff auf A_{jk} mittels `A(j,k)`. Stellen Sie mittels `assert` sicher, dass die Koeffizienten im zulässigen Bereich sind, d.h. $j, k \in \{1, \dots, n\}$ für $A \in \mathbb{R}^{n \times n}$. Beachten Sie, dass der Speichervektor in C++ intern mit $\ell = 0, \dots, n^2 - 1$ indiziert wird.

Aufgabe 7 (4 Punkte). Schreiben Sie die Methode `isLowerTriangular` der Klasse `Matrix`. Diese liefert als Rückgabe 1, falls $A \in \mathbb{R}^{n \times n}$ eine untere Dreiecksmatrix ist, und 0 anderenfalls.

Hinweis. Eine Matrix $A \in \mathbb{R}^{n \times n}$ ist eine untere Dreiecksmatrix, falls $A_{jk} = 0$ gilt für alle Indizes $j, k = 1, \ldots, n$ mit $k > j$.

Lösung zu Aufgabe 7.

Aufgabe 8 (4 Punkte). Schreiben Sie die Methode `norm` der Klasse `Matrix`, die für $A \in \mathbb{R}^{n\times n}$ die Zeilensummenorm

$$\|A\| := \max_{j=1,\ldots,n} \sum_{k=1}^{n} |A_{jk}|$$

berechnet.

Hinweis. Den Absolutbetrag eines `double`-Wertes berechnet die Funktion `fabs`.

Lösung zu Aufgabe 8.

Aufgabe 9 (4 Punkte). Überladen Sie den Operator `*`, sodass `C = A*B` für $A, B \in \mathbb{R}^{n\times n}$ das Produkt

$$C = AB \in \mathbb{R}^{n\times n} \quad \text{mit} \quad C_{j\ell} = \sum_{k=1}^{n} A_{jk} B_{k\ell} \quad \text{für alle } j, \ell \in \{1, \dots, n\}$$

berechnet und zurückgibt. Stellen Sie mittels `assert` sicher, dass A und B dieselbe Dimension haben.

Lösung zu Aufgabe 9.

Aufgabe 10 (2 Punkte). Welchen Aufwand hat Ihre Implementierung des Produktes $C = AB$ für $A, B \in \mathbb{R}^{n \times n}$? Falls die Funktion für $n = 10^3$ eine Laufzeit von 0.1 Sekunden hat, welche Laufzeit erwarten Sie für $n = 3 \cdot 10^4$?

Lösung zu Aufgabe 10.

Hinweis. Zusätzlich zur Klasse `Matrix` sei eine Klasse `Vector` gegeben zur Speicherung von Vektoren $x \in \mathbb{R}^n$. Neben Konstruktor, Kopierkonstruktor, Destruktor und Zuweisungsoperator gibt es eine Methode `size`, um die Dimension n auszulesen. Die Koeffizienten x_j des Vektors $x \in \mathbb{R}^n$ erhält man mittels `x(j)`, wobei der Index $j = 1, \dots, n$ im mathematisch üblichen Sinn verwendet wird. Sie müssen keine dieser Methoden implementieren, sondern dürfen diese voraussetzen!

```
class Vector {
private:
  int n;
  double* coeff;

public:
  Vector(int=0, double=0);
  Vector(const Vector&);
  ~Vector();
  Vector& operator=(const Vector&);

  int size() const;

  const double& operator()(int) const;
  double& operator()(int);
};
```

Aufgabe 11 (2 Punkte). Gegeben sei eine untere Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ (d.h. $A_{jk} = 0$ für $k > j$), die zusätzlich $A_{jj} \neq 0$ für alle $j = 1, \dots, n$ erfüllt. Sei $b \in \mathbb{R}^n$. Dann gibt es einen eindeutigen Vektor $x \in \mathbb{R}^n$ mit $Ax = b$. Leiten Sie mit Hilfe der Formel des Matrix-Vektor-Produktes

$$b_j = \sum_{k=1}^{n} A_{jk} x_k \quad \text{für } j = 1, \dots, n$$

eine Formel für die Koeffizienten x_j von $x \in \mathbb{R}^n$ her.

Lösung zu Aufgabe 11.

Aufgabe 12 (5 Punkte). Schreiben Sie eine Funktion `solveLowerTriangular`, die für einen Vektor $b \in \mathbb{R}^n$ und eine untere Dreiecksmatrix $A \in \mathbb{R}^{n \times n}$ mit $A_{jj} \neq 0$ für alle $j = 1, \dots, n$ die eindeutige Lösung $x \in \mathbb{R}^n$ des linearen Gleichungssystems $Ax = b$ berechnet und zurückgibt. Stellen Sie mittels `assert` sicher, dass A und b passende Dimension haben und dass A wirklich eine untere Dreiecksmatrix ist. Stellen Sie ferner mittels `assert` sicher, dass $A_{jj} \neq 0$ für alle $j = 1, \dots, n$ gilt.

Lösung zu Aufgabe 12.

Aufgabe 13 (5 Punkte). Was ist der Shell-Output des folgenden Programms? Was macht die `while`-Schleife bzw. was ist die mathematische Funktionalität im Code?

```cpp
#include <iostream>
#include <cassert>

using std::cout;
using std::endl;

class foobar {
private:
  int x;
  int y;
public:
  foobar(int x, int y) {
    assert(x>0);
    assert(y>0);
    this->x = x;
    this->y = y;
    cout << "new: x=" << x << ", y=" << y << endl;
  }
  ~foobar() {
    cout << "old: x=" << x << ", y=" << y << endl;
  }
  foobar(const foobar& eprog) {
    int a = eprog.x;
    int b = eprog.y;
    int c = 0;
    while ( a != b ) {
      if ( a < b ) {
        c = a;
        a = b;
        b = c;
      }
      a = a - b;
      cout << "a=" << a << ", b=" << b << ", c=" << c << endl;
    }
    x = eprog.x/a;
    y = eprog.y/b;
  }
};


int main() {
  foobar father(20,45);
  foobar son = father;
  return 0;
}
```

Lösung zu Aufgabe 13.