Dirk Praetorius,
Thomas Führer

Sommersemester 2014
26.05.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 11

Hinweis: *In dieser Serie benötigen Sie die Matrix-Klasse aus der letzten Übung.*

Aufgabe 11.1. Überladen Sie den Operator `+` für die Klasse `Matrix` aus der letzten Übungsserie. Schreiben Sie auch ein Programm in welchem Sie die Implementierung testen.

Aufgabe 11.2. Überladen Sie den Operator `*` (Matrix-Matrix Multiplikation) für die Klasse `Matrix` aus der letzten Übungsserie. Schreiben Sie auch ein Programm in welchem Sie die Implementierung testen.

Aufgabe 11.3. Überladen Sie den Operator `+` für die Klasse `MyVector` aus der VO. Schreiben Sie auch ein Programm in welchem Sie die Implementierung testen.

Aufgabe 11.4. Überladen Sie den Operator `*` für die Klasse `MyVector` aus der VO. Dabei sei die Multiplikation zweier Vektoren $x, y \in \mathbb{R}^n$ komponentenweise definiert, d.h die j-te Komponente des Produktvektors $z = x * y$ lautet $z_j = x_j y_j$. Schreiben Sie auch ein Programm in welchem Sie die Implementierung testen.

Aufgabe 11.5. 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 11.6. Überladen Sie die Funktion `print`, so dass

```
void print(vector<double> &d);
void print(vector<Complex> &c);
```

entweder einen Vektor mit `double`- oder `Complex`-Einträgen ausgibt. Testen Sie Ihre Implementierung entsprechend.

Aufgabe 11.7. Überladen Sie die Funktion `norm`, so dass

```
double norm(Matrix& A);
double norm(vector<double>& v);
```

entweder die Frobeniusnorm $\sqrt{\sum_{j=1}^m \sum_{k=1}^m A_{jk}^2}$ einer $m \times n$-Matrix A bzw. die ℓ_2-Norm $\sqrt{\sum_{j=1}^n v_j^2}$ eines Vektors $v \in \mathbb{R}^n$ zurückliefert. Testen Sie Ihre Implementierungen entsprechend. Warum ist es eine gute Idee für die Übergabe *Call by Reference* zu verwenden?

Aufgabe 11.8. Schreiben Sie zwei Funktionen

```
double prod(vector<double> &a, double c);
double prod(vector<double> &a, vector<double> &b, double c);
```

welche im ersten Fall $c \prod_{j=1}^n a_j^2$ und im zweiten Fall $c \prod_{j=1}^n a_j b_j$ zurückliefern sollen. Testen Sie die Implementierungen.