Prof. G. Zachmann
P. Dittmann
H. Meißenhelter

Universität Bremen
Fachbereich 3
Arbeitsgruppe CGVR
16. November 2020

Wintersemester 2020/21

Übungen zu Computergraphik - Blatt 2

Abgabe am 23.11.2020, 23:59 Uhr

Aufgabe 1 (C++-Programmierung von Matrizen, *7 + 2 Punkte*)

Implementieren Sie eine Klasse für $n \times m$-Matrizen in C++, wobei die einzelnen Elemente als Floating-Point-Zahlen dargestellt werden. In Stud.IP wurde das Framework „MatrixImplementation“ bereitgestellt, welches Sie um einige Features erweitern sollen. Es dürfen keine externen Bibliotheken zur Hilfe der Implementierung benutzt werden.

Die Klasse sollte folgende Funktionalität bereitstellen:

1. Einen Konstruktor der zwei Argumente n und m nimmt und dynamisch Speicher für $n \times m$ viele Elemente alloziert. ($\frac{1}{2}$ Punkte)
2. Einen Kopierkonstruktor, welcher als Argument eine andere Matrix nimmt und eine Kopie von dieser erstellt, d.h. die resultierende Matrix hat die selbe Ordnung und Elementwerte wie der Parameter. ($\frac{1}{2}$ Punkte)
3. Einen Dekonstruktor, der alle Ressourcen den Sie in Konstruktoren alloziert haben wieder freigibt. ($\frac{1}{2}$ Punkte)
4. Zugriffsoperator (i,j) für den Zugriff auf das Element in Zeile i und Spalte j. Zu Gunsten der Performance, soll der Operator nur im Debug-Modus überprüfen, ob die Koordinaten im legalen Bereich sind. Überprüft dazu per Preprocessor-Befehl #ifdef DEBUG ob das Define definiert ist. (1 Punkt)
5. Zuweisungsoperator =, der es dem Aufrufer erlaubt Ordnung und Inhalt von einer Matrix auf eine andere Matrix zu kopieren. (1 Punkt)
6. Addition mit einer anderen Matrix. ($\frac{1}{2}$ Punkte)
7. Subtraktion mit einer anderen Matrix. ($\frac{1}{2}$ Punkte)
8. Multiplikation mit einer anderen Matrix. ($1\frac{1}{2}$ Punkte)
9. Multiplikation mit einem Skalar. ($\frac{1}{2}$ Punkte)
10. Ausgabe der Matrix auf stdout. Die Ausgabe sollte so formatiert sein, dass man Ordnung und Elemente erkennen kann. ($\frac{1}{2}$ Punkte)

Es wird außerdem gefordert: Bei den Rechenoperationen soll immer auf kompatible Ordnungen geprüft werden. Im Fehlerfall soll eine Exception geworfen werden.

Bonus: Implementieren Sie eine Vektor-Klasse, welche eine variable Anzahl von Elementen in dynamisch alloziertem Speicher hält. Die Vektorklasse sollte das Skalar- und Kreuzprodukt mit anderen Vektoren unterstützen. Erweitern Sie außerdem Ihre Matrix-Klasse um Multiplikation mit einem Vektor von rechts, der Vektor soll als Spaltenvektor behandelt werden. Achten Sie auf die korrekte Dimensionierung der Vektoren bei allen Rechenoperationen.

Aufgabe 2 (Kreuzprodukt, *3 Punkte*)

Zeigen Sie, dass das Kreuzprodukt $\vec{v} \times \vec{w}$ in $\mathbb{R}^3$ orthogonal zu $\vec{v}$ ist.