

1. Übungsblatt zur Vorlesung Computergraphik im WS 2025/26

Besprechung am Montag, Dienstag, Mittwoch 10.11. ,11.11 & 12.11.2025

Hinweis:

Sie werden im Verlauf des Kurses Aufgabenblätter lösen, welche eine OpenGL 4.3-kompatible Umgebung voraussetzen. **Sollten Sie keinen Zugang zu einem geeigneten Computer haben, können Sie einen VIS-Account beantragen und die PCs in den VIS-/VISUS-Pools verwenden.**

Aufgabe 1 *Einfache Funktionen und Konsoleneingabe und -ausgabe [2 Votierpunkte]*

In dieser Aufgabe sollen Sie sich mit dem Schreiben einfacher Funktionen und der Ein- und Ausgabe über die Konsole vertraut machen. Das Programm soll vom Benutzer auf der Konsole den Radius eines Kreises abfragen, die Kreisfläche berechnen und sie auf der Konsole ausgeben. Das gegebene Programmskelett enthält bereits den Einstiegspunkt des Programms (die `main()`-Funktion) mit einer ersten Textausgabe auf der Konsole. Zuerst muss aber eine Konfigurationsdatei für CMake erstellt und das Projekt mit CMake konfiguriert werden.

1. Erstellen Sie die Datei `CMakeLists.txt` im Verzeichnis `Aufgabe1` und schreiben Sie eine Konfiguration, welche eine ausführbare Datei als Output generiert. Ziehen Sie hierfür die CMake-Dokumentation zu Rate: <https://cmake.org/documentation/>.
2. Implementieren Sie eine Funktion `circlearea`. Die Funktion soll die Kreisfläche für einen gegebenen Radius berechnen. Ergänzen Sie in der `main()`-Funktion das Einlesen von Eingaben für den Kreisradius auf der Konsole und die Ausgabe des berechneten Wertes.

Aufgabe 2 *Verwendung der Standard Template Library (STL) [3 Votierpunkte]*

In dieser Aufgabe sollen Sie sich mit der Verwendung einiger grundlegender STL-Container (`std::array`, `std::vector` und `std::list`) vertraut machen. Ergänzen Sie dazu das Struct `Point3D` und die Klasse `LineStrip3D` im gegebenen Programmskelett. Anschließend ergänzen Sie die `main()`-Funktion um ein paar einfache Tests Ihrer Implementierung. Auch für diese Aufgabe muss zunächst eine CMake-Konfigurationsdatei erstellt werden.

Aufgabe 2.1 `Point3D`

1. Ergänzen Sie das Struct `Point3D` um einen Container, der die drei Koordinaten eines Punktes im 3D Raum enthält. Verwenden Sie den für diesen Zweck am besten geeigneten STL-Container und begründen Sie Ihre Wahl.
2. Implementieren Sie den Constructor und die deklarierten Funktionen von `Point3D`.

Aufgabe 2.2 LineStrip3D

1. Ergänzen Sie die Klasse `LineStrip3D` um einen Container, der beliebig viele Punkte (jeweils vom Typ `Point3D`) eines Linienzuges aufnehmen kann. Verwenden Sie den für diesen Zweck am besten geeigneten STL-Container und begründen Sie Ihre Wahl.
2. Implementieren Sie die Funktion `addPoint`, die Funktion `removePoint` und die Funktion `computeLength` der Klasse `LineStrip3D`.

Aufgabe 2.3 Testprogram

1. Legen Sie in der `main()`-Funktion einen Container für beliebig viele Linienzüge an. Verwenden Sie den für diesen Zweck am besten geeigneten STL-Container und begründen Sie Ihre Wahl. (*Hinweis: Sie können davon ausgehen, dass hinzugefügte Objekte zur Laufzeit nicht mehr entfernt werden.*)
2. Fügen Sie dem Container mehrere Linienzüge hinzu. Geben Sie anschließend die Länge der einzelnen Linienzüge auf der Kommandozeile aus.
3. Sortieren Sie den Container nach der Länge der Linienzüge und geben Sie die Länge der einzelnen Linienzüge erneut auf der Kommandozeile aus, um das Ergebnis der Sortierung zu überprüfen. (*Hinweis: Verwenden Sie zum Sortieren eine Funktion aus der `algorithm`-Bibliothek*)

Aufgabe 3 Pointer und Resource Acquisition Is Initialization (RAII)-Konzept [2 Votierpkt.]

In dieser Aufgabe sollen Sie sich anhand einer einfachen Matrix-Implementierung mit Speicherallokation und dem Umgang mit Pointern, sowie der Umsetzung des RAII-Konzeptes vertraut machen. Dazu ist ein Programmskelett gegeben, in dem Sie die Klasse `MatrixInt` erweitern und verwenden sollen. Das Programm lädt zwei Matrizen aus Dateien, multipliziert diese und schreibt das Ergebnis anschließend in eine dritte Datei. Auch für diese Aufgabe muss zunächst eine Konfigurationsdatei erstellt werden.

1. Implementieren Sie den fehlenden Constructor und Destructor der `MatrixInt`-Klasse und ergänzen Sie den Multiplikationsoperator `MatrixInt operator*(...)`; so, dass eine Matrixmultiplikation durchgeführt werden kann.
Ergänzen Sie in der Funktion `loadMatrix` das Füllen der Matrix mit den Werten aus der geladenen Datei.
2. Warum wird in den Funktionen `loadMatrix` und `writeMatrix` der Parameter `Matrix` mit `&` bzw. `const&` versehen?
Nennen Sie einen Vorteil des RAII-Konzeptes.

Aufgabe 4 Funktionsdefinitionen [2 Votierpunkte]

In den vorgegebenen Sourcecodes hat sich ein Fehler eingeschlichen. Auch für diese Aufgabe muss zunächst eine Konfigurationsdatei erstellt werden.

1. Beheben Sie diesen Fehler.
2. Erklären Sie die Ursache des Problems und weitere Implikationen.

Aufgabe 5 CMake-Projekt [1 Votierpunkt]

Erstellen Sie eine CMake-Konfiguration, welche die gesamte Übung in einem Projekt zusammenfasst.