Praktikum: Allgemeine Hinweise

Ziel dieses Praktikums ist es, die in der Vorlesung erworbenen Kenntnisse zu vertiefen und praktisch anzuwenden, sowie Sie zum Selbststudium anzuleiten. Außerdem sollen Sie erste Erfahrungen sammeln im gemeinsamen Erstellen von Software: Man einigt sich auf eine Schnittstelle und jeder Beteiligte löst unabhängig vom Anderen einen Teil der Aufgabe. Die Praktikumsaufgaben sind daher in **Zweier-Teams** zu bearbeiten. Zur Software-Entwicklung gehört auch das **systematische Testen** und eine zielgerichtete **Fehlersuche**: Sind die einzelnen Komponenten fehlerhaft oder arbeiten sie nur nicht korrekt zusammen. Diese Kenntnisse werden ebenfalls im Rahmen dieses Praktikums vermittelt.

Alle Aufgaben sind zu Hause vorzubereiten, d.h. die Vorlesungsinhalte müssen nachgearbeitet werden und verstanden sein. Die Vorbereitung dient dazu, dass Sie sich kritisch mit den in der Vorlesung behandelten Inhalten beschäftigen und dass Sie nicht behandelte Themen selbständig erarbeiten. Eine gute Mitarbeit in den Übungen ist hilfreich.

Im Praktikum versuchen die Betreuer durch Fragen sicherzustellen, dass die behandelten Themen verstanden sind. Die korrekte Beantwortung der Fragen ist notwendig, um das Praktikum erfolgreich zu absolvieren. Zum Praktikumstermin dürfen Sie Bücher, Vorlesungsfolien sowie -mitschriften mitbringen, natürlich können Sie Informationen auch im Internet suchen und nachlesen. Die Aufteilung, wer aus der Zweiergruppe welche Teile bearbeitet, bleibt Ihnen überlassen. Notwendige Programmteile werden ggf. von uns bereitgestellt. Am Ende des Praktikums müssen alle Module zusammen kompilierbar sein und das Programm die gewünschte Funktionalität aufweisen. Vielleicht sind in den von uns bereitgestellten Programmteilen auch Fehler enthalten.

Im Laufe des Praktikums soll eine etwas umfangreichere Software entstehen. An jedem Praktikumstermin wird ein kleiner Teil der Software implementiert, sodass am Ende daraus ein echtes Produkt zusammengesetzt werden kann. Jeder Teil wird für sich getestet (Komponententest). Die Funktionalität wächst also im Laufe der Zeit. Man nennt dieses Vorgehen auch inkrementelle Software-Entwicklung.

Das Produkt, was wir entwickeln wollen, ist ein Routenplaner. Darin soll eine Karte mittels eines Graphen modelliert werden. Die Knoten repräsentieren die Städte und die Kanten die Strecken dazwischen. Weitere Funktionalitäten wie das Finden der kürzesten Wegstrecke, effiziente Möglichkeiten der Integration von Wegänderungen bei Stau und Sperrungen, sowie Streckenplanung mit mehreren Zielpunkten müssen zusätzlich implementiert werden.

Viele Softwareprojekte in der heutigen Zeit verwenden Software, die von Dritten programmiert wurde. Um die Einbindung fremder Software zu üben, wird das Praktikum ein externes Framework mit dem Namen SFML verwenden.

SFML

Simple and Fast Multimedia Library (SFML; deutsch: Einfache und schnelle Multimedia-Bibliothek) ist ein plattformunabhängiges, objektorientiertes Open-Source-Multimedia-Framework, das unter der zlib/libpng-Lizenz steht. Es ist in C++ geschrieben und greift intern auf betriebssystemspezifische Funktionen sowie externe Bibliotheken zurück. Neben C++ bietet es Anbindungen für die Programmiersprachen C, .NET, Python, etc.. SFML wurde mit der Intention entwickelt, möglichst benutzerfreundliche und effiziente Multimedia-Programmierung auf hohem Abstraktionslevel zu erlauben, daher auch der Name der Bibliothek. [1]

[1] siehe https://de.wikipedia.org/wiki/Simple_and_Fast_Multimedia_Library