TU Bergakademie Freiberg
Arbeitsgruppe Softwareentwicklung und Robotik
Prof. Dr. Sebastian Zug
M. Sc. Georg Jäger

Softwaretechnologie-Projekt (Prototyp)

Aufgabenblatt 1

Vorbereitungsaufgaben

1. Hello World!

Schreiben, kompilieren und führen Sie ein C++ Programm aus, dass auf der Konsole „Hello World!“ ausgibt. Welche Sprachkonstrukte müssen Sie nutzen? Erklärt die Funktion jeder einzelnen Quellcode-Zeile!

2. Grundlegende Syntax

Verschaffen Sie sich einen Überblick über die grundlegende Syntax von C++. Beantworten Sie dazu folgende Fragen:

- Welche primitiven Datentypen gibt es in C++? Wie viel Platz nehmen sie im Speicher ein?
- Was ist eine Referenz und was ist ein *Pointer*?
- Wie werden Funktionen und Klassen definiert?
- **Zusatz:** Was bedeuten die unterschiedlichen *Value Categories* in C++ *Expressions*?

3. Heap und Stack Allokationen

In C++ kann Speicher für Variablen entweder auf dem *Heap* oder dem *Stack* angelegt werden. Was ist der Unterschied? Welchen Vor- und Nachteil bringen beide Varianten mit sich? Illustrieren Sie den Unterschied mit Hilfe eines *double*-Arrays.

4. Rule of Five

Um die Nutzung von Klassen und Objekten in C++ zu vereinfachen, sollte man die *Rule of Five* beachten. Worum handelt es sich dabei? Schreiben Sie ein kurzes Beispielprogramm!

Übungsaufgaben

Um die Konzepte der Objekt-Orientierung in C++ besser zu verstehen, wollen wir verschiedene Klassen rund um das Thema „Geometrie“ implementieren. Nutzen Sie jeweils die Kommandozeile um sinnvolle Ausgaben zu machen.

1. Klasse „Punkt“

Implementieren Sie zunächst eine Klasse, die einen 2D Punkt repräsentiert. Wenden Sie dazu die *Rule of Five* an und implementieren Sie alle zugehörigen Funktionen explizit.

2. Klasse „Polygon“

Verwendeen Sie die Klasse „Punkt“ um ein Polygon (Vieleck) zu implementieren.

3. Klasse „Rechteck“

Leiten Sie von dem „Polygon“ ab um ein Rechteck zu implementieren. Versuchen Sie möglichst wenig redundanten Code zu schreiben.

4. Convex Hull - Rechteck

Schreiben Sie eine Funktion die eine List von Punkten als Eingabe bekommt und das minimale Rechteck berechnet, das alle Punkte umschließt.

5. Convex Hull - Polygon

Schreiben Sie eine Funktion die eine List von Punkten als Eingabe bekommt und das minimale Polygon berechnet, das alle Punkte umschließt.
Hinweis: Sie können weitere Klassen und Funktionen hinzufügen um den erforderlichen Quellcode für den Algorithmus gering zu halten.