5. Using Structs to structure related data

Die folgenden Aufgaben sind zur Vertiefung des erlernten Wissens gedacht. Bei Fragen meldet euch gerne per E-Mail.

Aufgabe 1: Formen

1. Erstelle structs für die Formen Rechteck, Kreis und Dreieck. Die structs sollen genügend Daten enthalten, um den Flächeninhalt der verschiedenen Objekte zu errechnen.
2. Erstelle in deiner `main()`-Funktion jeweils zwei Instanzen der jeweiligen structs mit unterschiedlichen Daten. Gib die Daten (z.B. mithilfe von `println!()`) in einer beispielhaften Darstellung aus.
3. Schreibe nun jeweils einen `impl`-Block pro struct mit einer `new()`-Methode. Verändere die Instanzierungen in `main()` so, dass diese neue Methode genutzt wird.
4. Schreibe eine weitere Methode `area()`, welche den Flächeninhalt der jeweiligen Datenstruktur als `f32` zurückgibt. Ändere die Ausgabe in `main()` so ab, dass zusätzlich zu den Formen auch ihre Größe ausgegeben wird.
5. Schreibe eine Funktion `from_rect(rect: Rectangle)` für den Kreis. Sie soll einen neuen Kreis mit dem Flächeninhalt des übergebenen Rechtecks erstellen. Erstelle in `main()` eine beispielhafte Instanz, welche diese neue Funktion nutzt.

Aufgabe 2: Binary Tree Nodes

Diese Aufgabe ist schwierig! Sie ist für diejenigen gedacht, welche schon etwas Erfahrung mit Rust haben oder eine kleine Herausforderung suchen.

Ein Binärbaum besteht aus einzelnen Knoten, sogenannten `Node`s.

1. Implementiere eine Node mithilfe von einer `struct` in Rust. Klappt dies so wie gedacht? Weswegen gibt es Probleme beim Compilen?
2. Ein Binärbaum besteht entweder aus Knoten (Nodes) oder Blättern (Nodes ohne Kinder). Versuche nun, dies in deinem `struct` zu implementieren. **Hinweis: Es gibt eine sehr elegante Lösung.**
3. Wie lässt sich das Problem aus 1 lösen? Versuche, einen beispielhaften Binärbaum in deiner `main()`-Funktion aufzubauen.