Computermathematik – Übungsblatt 8

- **Abgabeschluss:** Di 14. 1. um 23:59
- **Präsentation:** Mi 15. 1. in der Übungsgruppe
- **Abgabeformat:** `.sws` oder `.sagews`

Aufgabe 13 – Approximation des Goldenen Schnitts (4 Punkte)

Der goldene Schnitt $\phi \approx 1.618$ ist eine Zahl, die (in Form des Verhältnisses $\phi : 1$) eine wichtige Rolle in der Ästhetik und Architektur spielt, aber auch in der Natur wiederzufinden ist. Die Zahl ist irrational, kann aber mittels der Fibonacci-Folge F_n angenähert werden, da $\lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \phi$. (Zur Erinnerung: F_n ist rekursiv definiert durch $F_1 = F_2 = 1$ und $F_n = F_{n-1} + F_{n-2}$ für $n \geq 3$.)

Erstelle einen Generator, der die Approximation F_{n+1}/F_n für aufsteigende n zurückgibt. Der Generator soll auch einen optionalen Parameter für eine sinnvolle Abbruchbedingung anbieten, beispielsweise eine bestimmte Genauigkeit oder eine maximale Iterationszahl. Nutze diesen Generator, um das Konvergenzverhalten der Approximation bildlich zu veranschaulichen, etwa durch einen Plot des Approximationswerts als Funktion von n, oder eine Serie von Rechtecken mit dem approximierten goldenen Seitenverhältnis.

Aufgabe 14 – Polygone rotieren (4 Punkte)

Erstelle eine Funktion, die einen Punkt in der Ebene $P = (x, y)$ um einen Winkel α (in Radiant) rotiert (gegen den Uhrzeigersinn, um den Nullpunkt). Benutze diese Punkt-Rotation, um ganze Polygone zu drehen. Ein Polygon (Vieleck) besteht aus einer Liste von Punkten, die durch Linien miteinander verbunden sind. Das ganze Polygon wird gedreht, indem jeder einzelne Punkt der Liste rotiert wird (die Reihenfolge der Punkte in der Liste bleibt aber gleich). Teste die Funktion für einige Polygone und stelle Original und Rotation jeweils gemeinsam in einem Plot dar (Hint: `polygon()`, Beispiel-Output in Abbildung 1).

Hint: Eine mögliche, einfache Formel für die Berechnung so einer Punkt-Rotation ergibt sich, wenn man den Vektor $P = (x, y)$ stattdessen als komplexe Zahl $z = x+iy$ interpretiert und diese mit $\mathrm{e}^{i\cdot\alpha}$ multipliziert. Das Ergebnis ist wieder eine komplexe Zahl z', deren Real- und Imaginärteil die Koordinaten des gedrehten Punktes P' ergeben (Hint: `real()`, `imag()`). Beispiel:

$$P = (2, 1) \quad \to \quad z = 2 + 1 \cdot i \quad \to \quad z' = e^{i\frac{\pi}{2}} \cdot (2 + 1 \cdot i) = -1 + 2 \cdot i \quad \to \quad P' = (-1, 2).$$

Bonus – Einfache Vektorgrafik rotieren (2 Bonuspunkte)

Mit dieser Rotier-Methode für Vektoren lassen sich natürlich nicht nur Polygone drehen, sondern auch einfache Linien oder Kreise (wie?). Erstelle eine Funktion, die eine ganze „Vektorgrafik“ bestehend aus Listen von Linien, Polygonen und Kreisen (oder weiteren Dingen?) um den Nullpunkt rotiert (oder auch spiegelt?). Wie könnte man das Ergebnis sinnvoll in einen Rückgabewert verpacken? Demonstriere die Funktion an einer einfachen Beispielgrafik.

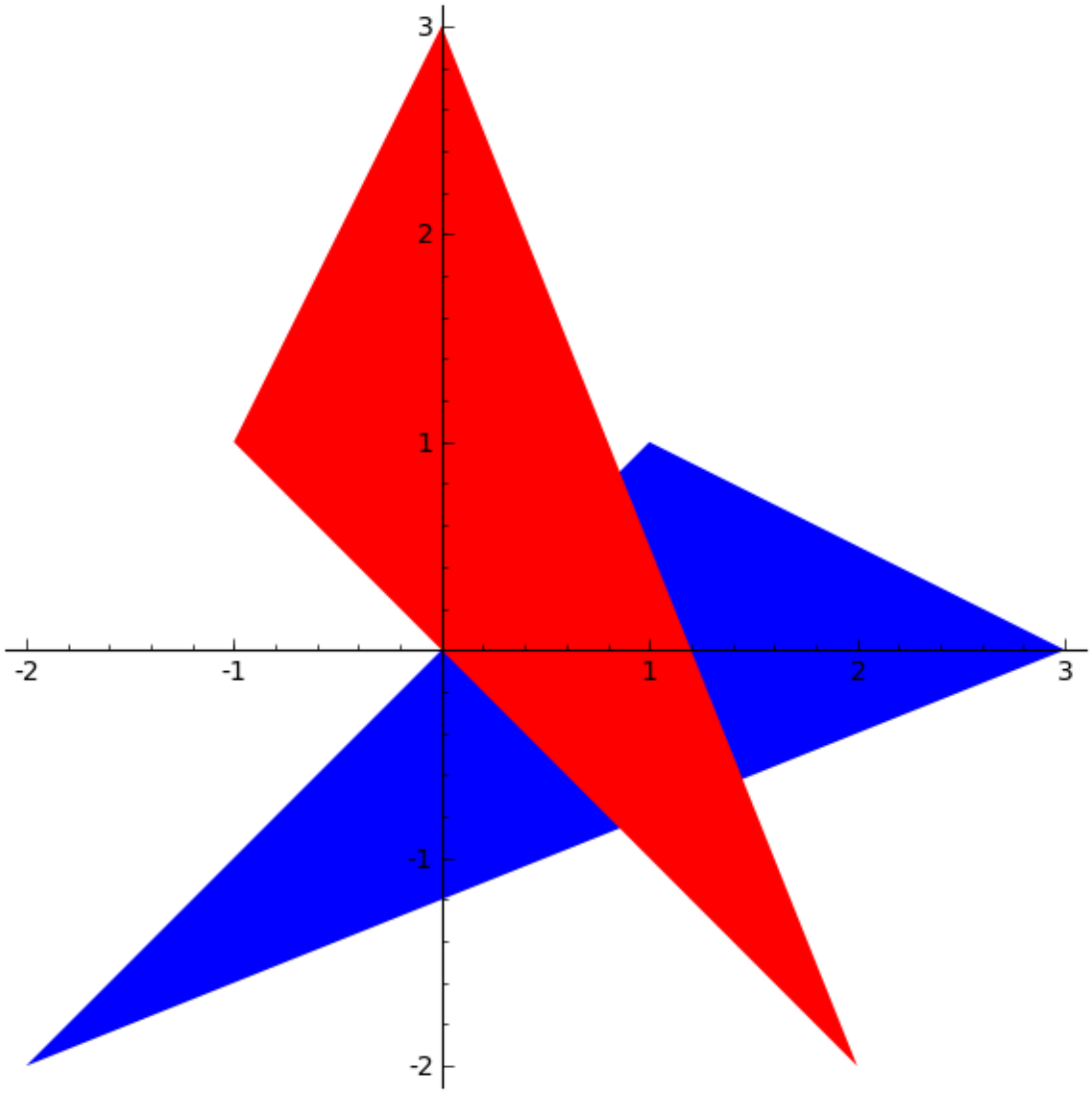

Abbildung 1: Drehung des blauen Polygons $L = [(3,0),(1,1),(-2,-2)]$ um den Winkel $\alpha = \frac{\pi}{2}$.