

Einführung in die wissenschaftliche Programmierung

Übungsblatt 8

1.) Komplexe Zahlen

Erstellen Sie eine Klasse `Komplex` um komplexe Zahlen darzustellen. Diese soll die Operationen $+$, $-$, \cdot , $/$ unterstützen, sowie eine Funktion für die Berechnung des Betrags und der komplexen Konjugation beinhalten. Können Sie die Klasse so ändern, dass man z.B. schreiben kann

```
>>> a = Komplex(1, 10)
>>> b = Komplex(1, 20)
>>> print a + b
2 + i30
>>> print abs(a)
10.0498756211
```

2.) Newton Verfahren

Mit dem Newton Verfahren kann eine Approximation der Lösung zu der Gleichung $f(x) = 0$ für eine stetig differenzierbare Funktion $f : \mathbb{R} \rightarrow \mathbb{R}$ bestimmt werden. Beschrieben wird das Verfahren über die Iterationsvorschrift

$$x_{n+1} = x_n - \frac{f(x)}{f'(x)},$$

wobei $x_0 \in \mathbb{R}$ gegeben ist.

Erstellen Sie zuerst eine Klasse `Ableitung` welche mit einer Funktion initialisiert wird. Die Klasse `Ableitung` soll dann eine Funktion mit Parameter $x \in \mathbb{R}$ implementieren, welche die Ableitung der übergebenen Funktion an der Stelle x ausgibt. Benutzen Sie für die Ableitung die einfache Approximation erster Ordnung

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}.$$

Schreiben Sie anschließend eine Funktion `newton` mit der Funktion `f`, der zugehörigen Ableitung `df`, einer Schranke `eps` und der maximalen Anzahl der Iterationen `N` als Parameter. Es soll dann bis zur Genauigkeit `eps` iteriert werden, jedoch maximal `N` viele Iterationen. Die Funktion `newton` soll die Nullstelle und den Wert an der Nullstelle (d.h. den Fehler) zurückgeben.

3.) Mandelbrot Menge

Die Mandelbrot-Menge `M` ist die Menge aller komplexen Zahlen `c`, für welche die Folge

$$z_{n+1} = z_n^2 + c, \quad z_0 = 0,$$

beschränkt bleibt. Visualisiert wird diese Menge üblicherweise wie in der Abbildung unten. Hierzu wird die Anzahl der Iterationen gezählt, bis **der Absolutwert der** Folge über z.B. den Wert 2.0 wächst. Wird 2.0 bereits nach z.B. 5 Iterationen erreicht, gibt man an der Stelle `c` in der komplexen Zahlenebene einen z.B. grauen Punkt aus. Benötigt man 10, 15 oder mehr Iterationen, dann z.B. einen grünen, blauen oder schwarzen Punkt.

Erzeugen Sie nun die Mandelbrot-Menge im Bereich $[-2, 0.5] \times [1, -1]$ und stellen Sie diese mit TKInter graphisch dar. Diskretisieren Sie dazu den Bereich in 50×50 Teilstücke und malen Sie für jeden dieser 50×50 Punkte ein Rechteck der Größe 4×4 . Folgendes könnte hilfreich sein:

- Haben Sie die TK Klasse mit `root = Tk()` instanziiert, können Sie mit `can = tk.Canvas(root, width=200, height=200)` eine "Leinwand" erzeugen.
- Auf dieser können Sie dann mit

```
can.create_rectangle(x, y, x + 4, y + 4, fill = "grey", outline = "grey")
```

ein 4×4 graues Rechteck (mit grauem Rand) an der Stelle (x, y) erzeugen.

- Sie müssen `root.update()` nach jedem Aufruf von `create_rectangle` aufrufen, um das Rechteck zu zeichnen.
- Verwenden Sie die Klasse `Komplex` von Aufgabe 1.

