MATHEMATISCHES INSTITUT
PROF. DR. ACHIM SCHÄDLE
MARINA FISCHER

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

25.10.2018

Computergestützte Mathematik zur Analysis – 3. Übungsblatt

Hinweis: Bearbeiten Sie das Blatt in JUPYTER. Gehen Sie dazu wie in Aufgabe 0 von Blatt 1 vor.

Aufgabe 9: (*Größter gemeinsamer Teiler*)
Befehle: `if`, `while`
Für zwei Zahlen a, $b \in \mathbb{Z}$ ist der größte gemeinsame Teiler (ggT) die größte natürliche Zahl N, die a und b ohne Rest teilt. Implementieren Sie zur Bestimmung des ggT von $a, b \in \mathbb{N}$...

(a) ... eine iterative Variante `ggt_it`(a, b):
Hierbei werden in der Funktion `ggt_it`(a, b) die drei Schritte

h = `Divisionsrest von` a/b
a = b
b = h

ausgeführt, solange $b \neq 0$ gilt. Sobald $b = 0$ erreicht ist, setzen wir $N = a$ und haben den ggT gefunden.

Zusatzfrage: Wie lässt sich die iterative Variante implementieren ohne die Hilfsvariable h zu verwenden?

(b) ... eine rekursive Variante `ggt_rek(`a`,`b`)`:
Falls $b = 0$ gilt, nimmt N den Wert a an. Andernfalls rufen wir die Funktion `ggt_rek` mit den Eingabeargumenten b und h erneut auf, wobei h der Divisionsrest von a/b ist, und speichern die Auswertung in der Variable N. Ist die Rekursion beendet, gibt N den ggT an.

(c) Testen Sie Ihre Funktionen mit den Zahlenpaaren (2469134, 8641969), (-345, 15), (7892389, -3).

Aufgabe 10: (*Eigene Klasse: Brüche*)
Floats können in PYTHON nicht so groß wie *Integer* werden, d.h. es gibt ganze Zahlen, die als *Integer*, aber nicht als *Floats* gespeichert werden können. Wir wollen daher eine Klasse `Bruch` entwickeln, die IMMER nur mit Integern rechnet und mit der man exakt mit Brüchen rechnen kann.
Ein (leeres) Grundgerüst für diese Klasse finden Sie auf der Internetseite zur Vorlesung.

(a) `Bruch` soll den Zähler und den Nenner als Integer übergeben bekommen und diese speichern. (Der Aufruf `a=Bruch(2,7)` entspricht also $2/7$, wobei uns `a.zaehler` den Wert 2 und `a.nenner` den Wert 7 ausgeben soll.) Der Nenner soll intern immer größer als 0 sein. Sollte er 0 sein, werden Zähler und Nenner auf 0 gesetzt und es wird eine sinnvolle Warnung ausgegeben.

(b) Um mit den Objekten der Klasse `Bruch` rechnen zu können, wollen wir die Standardsymbole $+$, $-$, $*$ und $/$ verwenden können. Dies geht durch das Implementieren bestimmter Methoden. Beispielsweise bekommt `__mul__(self, other)` (Multiplikation) sich selbst (`self`) und den anderen Faktor (`other`) übergeben und gibt ein Objekt der Klasse `Bruch` zurück.
D.h. für `a=Bruch(3,2)` und `b=Bruch(-5,9)` greift der Befehl `a*b` auf die Methode `a.__mul__(b)` zurück.
Achtung: Beim Dividieren kann es vorkommen, dass Sie durch 0 teilen (siehe (a)).

(c) Implementieren Sie die Klassenmethode `kuerzen(zaehler, nenner)`, welche den gekürzten Zähler und Nenner zurückgibt. Ihre Klasse soll jetzt immer den gekürzten Bruch speichern,

d.h für `a=Bruch(8,6)` soll `a.zaehler=4` und `a.nenner=3` sein.
Verwenden Sie hierfür entweder eines Ihrer eigenen Verfahren zur Bestimmung des ggT aus Aufg. 9 oder führen Sie außerhalb der Klasse den Befehl `from math import gcd` aus und verwenden die PYTHON-Implementierung `gcd`.

(d) Implementieren Sie die Klassenmethode `tofloat()`, welche den Bruch als Gleitkommazahl zurückgibt.

(e) Testen Sie Ihre Klasse an einigen Beispielen.

WICHTIG: Wie oben schon erwähnt, rechnen Sie in dieser Klasse IMMER nur mit Integern!

Aufgabe 11: (*Eigene Klasse: Drei- und Vierecke*)

(a) Ein achsenparalleles Rechteck lässt sich konstruieren, wenn wir den linken unteren Eckpunkt P_0 und den rechten oberen Eckpunkt P_1 gegeben haben. Schreiben Sie eine Klasse `Rectangle`, die die beiden Eckpunkte P_0 und P_1 jeweils als Tupel $(x_i, y_i),\ i = 0, 1$, übergeben bekommt und diese mit Hilfe der Methode `__init__` abspeichert. Vervollständigen Sie die Klasse mit Methoden, die den Flächeninhalt, den Umfang und den Mittelpunkt des Rechtecks ausgeben können.

(b) Testen Sie die Klasse `Rectangle` mit den Eckpunkten $P_0 = (0, 0)$ und $P_1 = (1, 1)$ bzw. $P_0 = (-1, 3)$ und $P_1 = (2, 7)$.

(c) Schreiben Sie eine Klasse `Triangle`, die die drei Eckpunkte P_0, P_1, P_2 eines Dreiecks erhält, wobei diese jeweils als Tupel aus den Koordinaten x_i und $y_i,\ i = 0, 1, 2$, gegeben sind. Speichern Sie die Eckpunkte in der Klasse ab. Implementieren Sie außerdem Methoden um den Flächeninhalt, den Umfang und den Schwerpunkt des Dreieckecks berechnen zu können.

(d) Testen Sie die Klasse `Triangle` mit den Eckpunkten $P_0 = (0, 0),\ P1 = (1, 0)$ und $P_2 = (0, 1)$ bzw. $P_0 = (1, 2),\ P1 = (3, 3)$ und $P_2 = (1.5, 4)$.

Aufgabe 12: (*Eigene Klasse: Quadratische Funktionen*)
Wir betrachten quadratische Funktionen der Form $f :\ \mathbb{C} \to \mathbb{C},\ x \mapsto a_0 + a_1 x + a_2 x^2$ mit $a_0, a_1, a_2 \in \mathbb{R}$. Erstellen Sie eine Klasse `Quadratic`, die über die folgenden Methoden verfügt:

(a) `Quadratic` soll die Koeffizienten a_0, a_1, a_2 übergeben bekommen und als Liste im Attribut `coeff` abspeichern. Außerdem möchten wir für ein Objekt `f` mit dem Befehl `f(x)` die Funktionsauswertung an der Stelle `x` erhalten. Implementieren Sie die dazugehörige Methode `__call__`.

(b) Wir wollen zwei quadratische Funktionen miteinander addieren oder voneinander subtrahieren können und daraus wieder Objekte der Klasse `Quadratic` erhalten. Implementieren Sie die entsprechenden Methoden.

(c) Schreiben Sie eine Methode `root()`, die eine Liste mit den, der Größe nach sortierten, reellen Nullstellen eines Objekts der Klasse `Quadratic` wiedergibt. Doppelte Nullstellen sollen hierbei nur einmal in der Liste vorkommen. Gibt es keine reelle Nullstelle, soll eine leere Liste ausgegeben werden.

(d) Wir wollen auch die erste Ableitung einer quadratischen Funktion berechnen können. Schreiben Sie hierzu eine Methode `diff()`, die ein Objekt der Klasse `Quadratic` wiedergibt.

Besprechung in den Übungen vom 29. Oktober - 9. November 2018.