Heinrich Heine Universität Düsseldorf

Mathematisches Institut
Prof. Dr. Achim Schädle
Melinda Hagedorn

31.10.2024

Computergestützte Mathematik zur linearen Algebra – 4. Übungsblatt

Aufgabe 13: (*Einführung Kontrollstrukturen*)

Tick, Trick und Track wollen an Halloween auf einer Straße mit 20 Häusern nach Süßigkeiten fragen. Sie gehen dafür die Häuser der Reihe nach ab und bekommen an jeder Tür maximal 5 Packungen Süßes. In ihren Taschen haben sie insgesamt Platz für 50 Packungen.
Simulieren Sie diese Situation mit Schleifen und `if`-Abfragen folgendermaßen:

- Die Personen in Haus X geben Y Packungen. Hier soll Y zufällig zwischen 0 und 5 sein (siehe Hinweis unten). Das wird durch folgende Ausgabe angezeigt:
  Die Personen in Haus X geben Y Packungen Süßes.
  Bei *einer* Packung wird
  Die Personen in Haus X geben eine Packung Süßes.
  ausgegeben. Sollten an einem Haus keine Süßigkeiten verteilt werden, so wird stattdessen ausgegeben:
  Die Personen in Haus X verteilen keine Süßigkeiten.
- Achten Sie hierbei darauf, dass man nur so viele Packungen bekommen kann wie in die Taschen passen.
- Wenn ihre Taschen voll sind, hören Tick, Trick und Track auf. Dann wird ausgegeben:
  Mehr Süßigkeiten können nicht getragen werden!
  Ansonsten soll
  Es passen noch X Packungen in die Taschen.
  ausgegeben werden. Achten Sie auch hier wieder auf den Sonderfall mit einer Packung.

Hinweis: Führen Sie den Befehl `from numpy.random import randint` am Anfang einmal aus. Mit `randint(6)` erhalten Sie dann eine ganze Zufallszahl zwischen 0 und 5 (gleichverteilt in $[0, 6)$).

Aufgabe 14: *(Klasse Quader)*

Ein Euler-Ziegel ist eine Quader mit ganzzahligen Kantenlängen und ganzzahligen Flächendiagonalen. Ein Euler-Ziegel ist primitiv, wenn die drei Kantenlängen keinen gemeinsamen Teiler ungleich 1 haben.

(a) Schreiben Sie eine Klasse `Quader`, der als Eingabeparameter drei natürliche Zahlen, die Höhe, Breite und Länge, übergeben werden. Die Klasse `Quader` soll Methoden zur Berechnung des Volumens, und der Oberfläche haben.

(b) Ergänzen Sie die Klasse `Quader`, so dass auch die Flächendiagonalen berechnet werden. Weiter soll `Quader` testen, ob es sich um einen primitiven Eulerziegel handelt.

Sind die Quader mit den Kantenlängen $(44, 117, 240)$ und $(132, 351, 720)$ primitive Euler-Ziegel?

(c) Leiten Sie von der Klasse `Quader` eine Klasse `Würfel` ab.

b.w.

Aufgabe 15: (*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 die sich zur exakten Berechnung von Brüchen eignet.
Ein (leeres) Grundgerüst für diese Klasse finden Sie auf der Homepage der 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. 10 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 16: (*Klasse: Polynomring*)
In der Vorlesung (Lektion 4) haben Sie eine Klasse `PolyGruppe` kennengelernt in der auf der Menge der Polynome $R[X]$ über dem Ring R ein Verknüpfung $+$ definiert wurde, so dass eine additive Gruppe $(R[X], +)$ entsteht. Die Elemente von $(R[X], +)$ sind in der Klasse `Polynom` definiert und werden hier als Dictionary mit den Koeffizienten $a_0, \ldots, a_n$ eines Polynoms $p(X) = \sum_{i=0}^{n} a_i X^i$, $n \in \mathbb{N}$, abgespeichert. Als Ring R sind nur die ganzen Zahlen, die reellen Zahlen und die komplexen Zahlen vorgesehen.

In der Datei `compla_poly_20241031` ist bereits das Gerüst für eine Klasse `PolyRing` angelegt. Ergänzen Sie diese, sodass auf der Menge der Polynome zusätzlich eine Verknüpfung $*$ definert wird, mit sich die Multiplikation von Polynomen berechnen lässt.

Besprechung in den Übungen vom 04.11.2024 bis 08.11.2024.