MATHEMATISCHES INSTITUT
PROF. DR. CHRISTIANE HELZEL
ANDREAS TROLL

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

18.1.2018

Computergestützte Mathematik zur linearen Algebra – 13. Übungsblatt

Aufgabe 46: (*Eigene Klasse: Chemische Elemente*)

(a) Erstellen Sie eine PYTHON-Klasse `Element`, die zur Initialisierung folgende Parameter übergeben bekommt: Name (Name des Elementes als String), Symbol (Symbol als String, Gold → 'Au'), Ordnungszahl (als Integer), Schmelzpunkt (in °C als Float), Siedepunkt (in °C als Float) und Temperatur (aktuelle Temperatur des Objektes in °C als Float). Sollte keine Temperatur übergeben werden, wird diese auf 20°C gesetzt.

(b) Zusätzlich soll das Attribut `Aggregatzustand` ('fest', 'flüssig' oder 'gasförmig' als String), basierend auf der übergebenen Temperatur und dem Schmelz- und Siedepunkt, bei der Initialisierung erzeugt (aber NICHT übergeben) werden.

(c) Schreiben Sie eine Klassenmethode `aendereTemperatur(temp)`, welche die Temperatur des Objektes auf `temp` ändert (der Aggregatzustand muss dabei ggf. aktualisiert werden).

(d) Der untere Grenzwert für die Temperatur beträgt -273,15°C. Erweitern Sie Ihre Klasse so, dass kleinere übergebene Temperaturen auf den absoluten Nullpunkt gesetzt werden.

(e) Schreiben Sie eine Klassenmethode `eigenschaften()`, welche eine kurze Zusammenfassung des Objektes (Name, Symbol, Temperatur und Aggregatzustand) ausgibt, beispielsweise so:
```
Name: Gold (Au)
Temperatur: 42°C (fest)
```

(f) Testen Sie Ihre Klasse an den Elementen Stickstoff (25°C), Erbium (-5,3°C) und Dysprosium. Lassen Sie sich die Eigenschaften ausgeben. Ändern Sie die Temperatur des Stickstoffs auf -200,02°C, die des Dysprosiums auf -314°C und die des Erbiums auf 3456,7°C. Geben Sie dann noch einmal die Eigenschaften aus.

Hinweis: Sie dürfen sich in Ihrer Klasse Hilfsfunktionen schreiben. Um auf diese innerhalb der Klasse zugreifen zu können, müssen Sie diese, wie bei Attributen, mit `self.methode(...)` aufrufen.

Aufgabe 47: (*Eigene Klasse: Brüche*)

Wie Sie schon gesehen haben, können *Floats* in PYTHON, im Gegensatz zu *Integern*, nicht beliebig groß werden. Außerdem treten beim Rechnen Rundungsfehler auf. Wir wollen nun eine Klasse `Bruch` entwickeln, 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 (`a=Bruch(2,7)` wäre also $\frac{2}{7}$). Hierbei soll der Nenner intern immer größer als 0 sein. Sollte er 0 sein, werden Zähler und Nenner auf 0 gesetzt und es wird eine Warnung ausgegeben.

(b) Um mit den Objekten rechnen zu können, wäre ist praktisch wenn wir die Standardsymbole $+$, $-$, $*$ und $/$ verwenden könnten. 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.
Also: `a=Bruch(3,2)`, `b=Bruch(-5,9)`: `a*b` ⇔ `a.__mul__(b)`
Achtung: Beim Dividieren kann es vorkommen, dass Sie durch 0 teilen (siehe a)).

(c) Implementieren Sie die Klassenmethode `kürzen(zaehler, nenner)`, welche den gekürzten Zähler und Nenner zurückgibt (siehe Aufg. 48. Das `math`-Modul hat aber auch eine Methode). Ihre Klasse soll jetzt immer den gekürzten Bruch speichern!

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

(e) Testen Sie Ihre Klasse an einigen Beispielen.

WICHTIG: Rechnen Sie in Ihrer Klasse IMMER nur mit Integern!

Aufgabe 48: (*Größter gemeinsamer Teiler*)
Seien $a, b \in \mathbb{Z}$. Der größte gemeinsame Teiler (ggT) ist die größte natürliche Zahl N die a und b ohne Rest teilt. Implementieren Sie...

(a) ...diese iterative Variante:

```
ggt_it(a,b) (mit a,b ∈ ℕ)
    solange b ≠ 0
        h=Divisionsrest von a/b
        a=b
        b=h
     N=a
```

OHNE die Hilfsvariable h zu verwenden (*Tipp:* Vorlesung 9).

(b) ...diese rekursive Variante:

```
ggt_rek(a,b) (mit a,b ∈ ℕ)
    falls b=0 dann
        N=a
    ansonsten
        N=ggt_rek(b,Divisionsrest von a/b)
```

in so wenig Zeilen wie Sie können (es geht mit einer Zeile).

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

Aufgabe 49: (Eigene Klasse: Student)

(a) Erstellen Sie eine PYTHON-Klasse `Student`, die zur Initialisierung folgende Parameter übergeben bekommt: Vorname (als String), Nachname (als String) und Matrikelnummer (als Integer). Zusätzlich soll das Attribut `punkte_in_compla` bei der Initialisierung als `None` erzeugt werden.

(b) Die Datei *Studentenliste.txt* (siehe Vorlesungsseite) enthält Nach- und Vornamen, die Matrikelnummer und die erreichte Punktzahl in der CompLA-Klausur einiger Studenten. Lesen Sie diese Datei ein und speichern Sie die Daten in einer Liste mit `Student`objekten.

(c) Schreiben Sie eine Methode `noten_ausgeben(liste)`, welche die Liste aus b) übergeben bekommt, aus der Punktzahl die Note bestimmt und die Informationen zeilenweise ausgibt (Name, Matr.nr, Note). Am Ende soll die Durchschnittsnote aller Teilnehmer ausgegeben werden:

```
Liste der Noten für CompLA:
Tick Duck (123456): 1
Trick Duck (123457): 1
...
Donald Duck (123452): 4
Durchschnittsnote: 1.37
```

Notenschlüssel:
p: Punkte, $p < 60$: 5.0, $60 \leq p < 70$: 4.0, $70 \leq p < 80$: 3.0, $80 \leq p < 90$: 2.0, $90 \leq p$: 1.0

Besprechung in den Übungen vom 22.1.-26.1.2018.