Programmieren in der Physik – PHY.A80 – SS 2020

Hausübungen 18. März 2020

Loops and Lists.

H4: Spektrallinien im Wasserstoffatom (1 P)

Ohne Berücksichtigung relativistischer Korrekturen werden die Energien $h\nu_{if}$ von emittierten Photonen im Wasserstoffatom durch folgende Beziehung beschrieben:

$$h\nu_{if} = -\frac{m_e e^4}{8\epsilon_0^2 h^2}\left(\frac{1}{n_i^2} - \frac{1}{n_f^2}\right)$$

Hierbei ist $m_e = 9.1094 \cdot 10^{-31}$ kg die Elektronenmasse, $e = 1.6022 \cdot 10^{-19}$ C die Elementarladung, $\epsilon_0 = 8.8542 \cdot 10^{-12}$ $\mathrm{C^2 s^2 kg^{-1} m^{-3}}$ die Dielektrizitätskonstante, und $h = 6.6261 \cdot 10^{-34}$ Js das Planck'sche Wirkungsquantum.

Schreibe ein Python-Programm `hydrogen.py`, das die zweifach verschachtelten Listen `Lyman` und `Balmer` erzeugt, die die Übergangsenergien (in Elektronenvolt) und die Wellenlängen $\lambda = \frac{c}{\nu}$ (in Namometer) (mit $c = 2.99792 \cdot 10^8$ m/s) für die Lyman-Serie ($n_f = 1$) und Balmer-Serie ($n_f = 2$) für je 5 Übergänge ($n_i = n_f + 1, n_f + 2, \cdots$) enthalten soll und diese auf dem Bildschirm in folgender Form ausgibt:

```
Lyman:
nf  ni  E(eV) lambda(nm)
1   2   xx.xx  xxx.x
1   3   xx.xx  xxx.x
1   4   xx.xx  xxx.x
1   5   xx.xx  xxx.x
1   6   xx.xx  xxx.x
Balmer:
nf  ni  E(eV) lambda(nm)
2   3   xx.xx  xxx.x
2   4   xx.xx  xxx.x
2   5   xx.xx  xxx.x
2   6   xx.xx  xxx.x
2   7   xx.xx  xxx.x
```

H5: Vektorprodukt über epsilon-Tensor (1 P)

Das Vektorprodukt zweier Vektoren $\vec{A}$ und $\vec{B}$ lässt sich mit Hilfe des ε-Tensors auf folgende Weise definieren:

$$C_i = \{\vec{A} \times \vec{B}\}_i = \varepsilon_{ijk} A_j B_k = \sum_{j=1}^{3}\sum_{k=1}^{3} \varepsilon_{ijk} A_j B_k$$

Schreibe ein Python-Programm `cross.py`, das das Vektorprodukt der beiden Vektoren $A_j = (1, 2, 3)$, $B_k = (-2, 3, -1)$ nach obiger Formel berechnet. Implementiere dabei die Vektoren $\vec{A}$ und $\vec{B}$ und deren Vektorprodukt $\vec{C}$ als Listen und den ε-Tensor als 3-fach verschachtelte Liste, und gib das Ergebnis durch einen einfachen print-Befehl `print(C)` aus.

H6: Daten fitten (0.5 P)

Gegeben sind die N Messpunkte (x_i, y_i):

```
xi = 0.12,  1.34,  3.45,  -1.20,  2.89,  3.23,  3.99,  -2.1, -3.56, 5.55
yi = 0.23,  0.89,  2.89,  -1.50,   3.45, 3.55,  4.01,  -1.50,  -2.89, 5.34
```

Diese sollen durch die Gerade $y = k \cdot x$ gefitted werden. Schreibe dazu ein Pythonprogramm `fit.py`, das die beste Steigung k der Ausgleichgeraden nach folgender Formel berechnet

$$k = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{n} x_i^2},$$

und das Ergbnis in der folgenden Form ausgibt:

```
Die beste Steigung ist k =  x.xxxxx
```

H7: Trapezverfahren (0.5 P)

Schreibe ein Pythonprogramm `trapez.py`, das das bestimmte Integral I der Funktion $f(x) = e^{-x^2}$ zwischen den Grenzen $a = 0$ und $b = 2$ nach folgender Formel näherungsweise berechnet (Trapezmethode):

$$I \approx \frac{\Delta x}{2}\left[f(a) + f(b)\right] + \Delta x \sum_{i=1}^{N-1} f(x_i)$$

Hierbei ist N eine Integer-Zahl, die die Anzahl der äquidistanten Subintervalle bezeichnet. Daher gilt für die Stützstellen x_i:

$$x_i = a + i \cdot \Delta x, \qquad \Delta x = \frac{b - a}{N}$$

Schreibe die numerischen Näherungswerte für I für $N = 10, 20, 50, 100, 500$ in die Liste `I_trapez` und gib diese durch ein einfaches `print(I_trapez)` am Bildschirm aus.

Hinweis: Sie können durch Abgeben der Hausübungen Bonuspunkte sammeln. Laden Sie dazu Ihre Lösungen in `moodle.uni-graz.at` hoch und beachten Sie die Abgabefrist: 31. März 2020, 16:00! Versehen Sie Ihr Programm mit Kommentaren und schreiben Sie Ihren Namen und Matrikelnummer als Kommentarzeile zu Beginn Ihrer Programme.