| Name, Vorname, Unterschrift: | Matrikelnummer: |

1 Autorennen (3 + 3.5 + 4.5 = 11 Punkte)

Eine Physikerin beobachtet die Geschwindigkeit von zwei Autos in einem Formel 1 Rennen. Sie stellt fest, dass das erste Auto mit einer Geschwindigkeit fährt, die gegeben ist durch die Formel $v_1 = 2.0 \cdot a_h \cdot t$, wobei $a_h = 100.0$ m/s^2 ist. Das zweite Auto fährt mit einer Geschwindigket gegeben durch $v_2 = v_0 + c_h \cdot t^2$, mit $v_0 = 80.0$ m/s und $c_h = 50.0$ m/s^3.

a) Schreiben Sie eine Python Funktion `calculateSpeeds(t)`, welche einen float `t` als Argument bekommt und die Geschwindigkeit der zwei Autos berechnet und zurückgibt.

b) Schreiben Sie eine Python Funktion `compareSpeeds(t)`, welche einen float `t` als Argument bekommt und die Funktion `calculateSpeeds` aufruft. Die Funktion soll den boolschen Wert `True` zurückgeben, falls das erste Auto schneller als das zweite ist und die Geschwindigkeit des zweiten Autos positiv ist. Ansonsten muss die Funktion den Wert `False` zurückgeben.

c) Schreiben Sie Python Code, der die Funktion `compareSpeeds` für jede der ersten 100 Millisekunden des Rennens, also für $t = 0, 0.01, 0.02, \ldots, 0.98, 0.99$, aufruft. Zu jeder Millisekunde soll auf der Kommandozeile ausgegeben werden, welches Auto gerade schneller fährt. Zum Beispiel: `t = 0.12: Auto 2 ist schneller!`

2 Objektorientierte Programmierung: Partikel (3 + 3.5 + 2.5 + 1 = 10 Punkte)

a) Jedes Teilchen hat einen Schwerpunkt $\vec{r} = (x, y)^T$ und eine Masse m. Schreiben Sie eine Klasse `Particle`, welche alle relevanten Größen als Membervariablen `x,y,m` hat. Diese sollen im Konstrukor über Argumente gesetzt werden. Alle Argumente werden als Skalare übergeben.

b) Schreiben Sie nun eine Methode für die Klasse `Particle`, welche den Operator `+` überlädt, sodass die Addition zweier beliebiger Partikel einen neuen `particle` zurückgibt, dessen Ort der Schwerpunkt $\vec{r}_s = (x_s, y_s)^T$ beider Partikel ist und dessen Masse die Summe beider Massen ist. Der Schwerpunkt ist über

$$\vec{r}_s = \frac{\sum_i m_i \vec{r}_i}{\sum_i m_i}$$

gegeben.

c) Definieren Sie nun die abgeleitete Klasse `CompositeParticle`, welche zusammengesetzte Partikel über ihren Schwerpunkt beschreibt und ihre Gesamtmasse beschreibt. Der Konstruktor soll als Argumente eine Liste von `Particle` Objekten (`list_of_particles`) erhalten. Diese Liste soll als Membervariable abgespeichert werden.

d) Instantiieren Sie nun ein Objekt `CompositeParticle`, welches aus den drei Partikeln

i	x_i	y_i	m_i
1	0	0	1
2	1	0	2
3	1	2	1.25

besteht.

3 Numpy Arrays und Matrizen (1.5 + 1 + 2 + 3 + 2 + 0.5 = 10 Punkte)

a) Welche Zahlen werden durch folgenden Code auf die Kommandozeile ausgegeben?

```
import numpy as np

A = np.array(  [[1,3, 5, 7], [2,4, 6, 8], [2,6,10,14], [4,8,12,16]])
print A[::2,1::2]
```

b) Nennen Sie einen Vorteil von `numpy` Arrays im Vergleich zu Standard-Listen in python!

c) Wie lautet der einzeilige Befehl, um all diejenigen Elemente der Matrix A aus Teilaufgabe a) mit minus Eins zu multiplizieren, die größer als 5 sind?

d) Ein Programm zur Simulation von Tragflaechenumströmungen berechnet zwei Numpy Vektoren `fx` und `fy`, die die Kraefte der Strömung auf die Tragflaeche in x- und y-Richtung zu n verschiedenen Zeitpunkten speichern.
Geben Sie python Code an, um eine Textdatei `forces.txt` zu erstellen, die den Vektor `fx` als erste und `fy` als zweite Spalte—getrennt durch ein Leerzeichen—enthaelt!

e) Implementieren Sie die Generierung folgender Matrix B als `numpy` Array mithilfe der `numpy` Funktion `fromfunction()` und der Anonymen Funktionen (Lambda-Kalkül):

$$B = \begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix}$$

Tipp: Überlegen Sie, wie die Zahlenwerte der Matrix mit den Zeilen- bzw. Spaltenindices zusammenhängen.

f) Wie lautet der Befehl, um nur die Funktion `fromfunction()` aus dem `numpy` Paket zu importieren?

4 Dictionaries (3.5 + 3.5 + 1 + 1 + 2 = 11 Punkte)

Gegeben sei eine input Liste:
`l=[('Alice',4.2),('Bob',1.2),('Alice',6.32),('Charlie',8.01),...]`
Die Namen können doppelt vorkommen und die zugeordneten Zahlen sind alle positiv.

a) Schreiben Sie eine Funktion `sums(l)`, welche diese Art Liste als Argument übergeben bekommt und ein Dictionary zurückgibt, bei dem die Namen die keys sind und die Summe der Zahlen die Werte.

b) Schreiben Sie nun eine Funktion `find_max(l)`, welche diese Funktion auf `l` anwendet und dann den maximalen Wert und den dazugehörigen Namen zurückgibt.

c) Schreiben Sie eine Funktion `count(l)` welche die Häufigkeiten der Namen zählt und in einem Dictionary zurückgibt.

d) Wie kann man nun den häufigsten Namen in `l` unter Nutzung der zuvor definierten Funktionen in einem Einzeiler ausgeben?

e) Schreiben Sie eine Funktion `avg(l)`, welche den Durschnittswert für jeden Namen in einem Dictionary zurückgibt. Nutzen Sie dafür die zuvor geschriebenen Funktionen.