BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Prof. Dr.-Ing. Kristian Hildebrand

Visual and Scientific Computing
Python und Numpy

WiSe 2020/21

Das Lernziel dieser Übungsaufgabe ist, mit der Programmiersprache Python und der Bibliothek Numpy vertraut zu werden, um diese dann als Werkzeug in den weiteren Übungen einzusetzen. Machen Sie sich deshalb bitte so gut es geht damit vertraut und nutzen Sie dafür auch bitte Online-Resourcen wie https://docs.python.org/3/ und https://docs.scipy.org/doc/numpy/reference/.

Ihnen ist es selbst überlassen, ob Sie die Aufgaben in den vorgegebenen Dateien (*mergesort.py* und *numpy_basic.py*) oder in dem vorgegebenen Notebook (*01-basics.ipynb*) bearbeiten. Wenn Sie sich für den letzteren Fall entschieden haben, dann starten Sie einfach die *Jupyter Notebook App* in dem Verzeichnis in dem *01-basics.ipynb* liegt. Hilfe finden Sie hier: https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/execute.html.

Wichtiger Hinweis für alle Übungsaufgaben: Bitte geben Sie jede Übung so ab, dass die Aufgaben direkt ausführbar sind und die Ergebnisse mittels *print* ausgegeben werden. Andernfalls behalte ich mir Punktabzüge vor.

Aufgabe 1: Python Einführung (4 Punkte)

Um sich mit der Programmiersprache Python vertraut zu machen, implementieren Sie einen einfachen Mergesort-Algorithmus. Sie können dafür die vorgegebene Datei *mergesort.py* nutzen. *Nicht wundern, drei doppelte Anführungszeichen dienen als mehrzeiliger Kommentar. In diesem Fall auch* docstring *genannt. Das ist nur eine Beschreibung was die Funktion macht, welche Parameter sie bekommt und welche Rückgabewerte sie gibt.* Ein beispielhafte Eingabe und Ausgabe könnte so aussehen:

```
Eingabe: [ 5 14 10  7  2 13  1 12  6  9  0  3  8 11  4]
Ausgabe: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
```

Aufgabe 2: Numpy Einführung (5 Punkte)

Die Bibliothek *Numpy* ist sehr umfangreich und wir werden auch nicht jede Funktionalität brauchen. Es ist aber wichtig mit den grundlegenden Datenstrukturen vertraut zu sein. Ich habe hier eine Liste von kleinen Aufgaben zusammengestellt, die Sie in der Datei *numpy_basic.py* beantworten sollen. **Suchen Sie sich insgesamt 14 (von 23) Sterne zusammen und implementieren Sie diese**. Das ist häufig ein Einzeiler oder Zweizeiler. Die Musterlösung benötigt in jeder Teilaufgabe maximal drei Zeilen. Es muss aber nicht krampfhaft versucht werden diese Vorgabe zu erreichen, dies soll nur ein Hinweis darauf sein, dass Sie keinen langen Code schreiben müssen. Lesen Sie sich selbstständig in die Dokumentation ein und versuchen Sie die Fragen mit den notwendigen Numpy-Funktionen zu beantworten. Sollten Sie keine Funktion finden, implementieren Sie eine Lösung selbst in einer kleinen Funktion. Sie können auch im Kursforum nach Hilfe suchen. **Bitte geben Sie die Lösung jeweils auf der Konsole aus (mittels print), sonst werde ich Punkte abziehen**. Hinweis: die print Anweisung zählt nicht zu dem Einzeiler oder Zweizeiler, wenn im Hinweis also steht geht in einer Zeile muss in dieser nicht auch noch die print Anweisung kommen.

a) (*) Erzeugen Sie einen Vektor a mit Nullen der Länge 10 (10 Elemente) und setzen den Wert des 5. Elementes auf eine 1.

b) (*) Erzeugen Sie einen Vektor b mit Ganzahl-Werten von 10 bis 49 (geht in einer Zeile).

c) (*) Erzeugen Sie einen Vektor mit 8 Einträgen zwischen -1 und 1 bei dem alle Werte die gleichen Abstände habe und sowohl -1 als auch 1 enthalten sind (geht in einer Zeile).

d) (*) Geben Sie nur das Stück (slice) von Vektor b) aus, das die Zahlen 21 bis 38 (Stellen 11 bis 28) beinhaltet (geht in einer Zeile).

e) (*) Ändern Sie den Vektor b) indem sie das Stück (slice) von Stelle 15 bis einschließlich Stelle 25 mit den Werten negierten Werten von Stelle 1 bis einschließlich Stelle 11 überschreiben (geht in einer Zeile).

f) (*) Drehen Sie die Werte des Vektors aus b) oder a) um (geht in einer Zeile).

g) (*) Summieren Sie alle Werte in einem Array (geht in einer Zeile).

h) (*) Erzeugen Sie eine 4x4 Matrix mit den Werte 0 (links oben) bis 15 (rechts unten) (geht in einer Zeile).

i) (*) Erzeugen Sie eine 5x3 Matrix mit Zufallswerteintegers zwischen 0-100 (geht in einer Zeile).

j) (*) Multiplizieren Sie eine 4x3 Matrix mit einer 3x2 Matrix (geht zwar in einer Zeile, aber benutzen Sie lieber Hilfsvariablen und drei Zeilen).

k) (*) Erzeugen Sie eine 5x5 Matrix und geben Sie jeweils die geraden und die ungeraden Zeile aus (geht jeweils in einer Zeile).

l) (**) Erzeuge eine 5x5 Matrix mit Zufallswerteintegers zwischen 0-100 und finde deren Maximum und Minimum und normalisieren Sie die Werte (sodass alle Werte zwischen 0 und 1 liegen - ein Wert wird 1 (max) sein und einer 0 (min)).

m) (**) Extrahieren Sie den Integer-Anteil eine Arrays von zufälliger Zahlen zwischen 0-10 auf 3 verschiedene Arten.

n) (**) Erzeugen Sie eine Matrix M der Größe 4x3 und einen Vektor v mit Länge 3. Multiplizieren Sie jeden Spalteneintrag aus mit der kompletten Spalte aus M. Nutzen Sie dafür Broadcasting.

o) (***) Erzeugen Sie einen Zufallsmatrix der Größe 6x2, die Sie als Kartesische Koordinaten interpretieren können ([[x0, y0],[x1, y1],[x2, y2]]). Konvertieren Sie diese in Polarkoordinaten.

p) (***) Erzeugen Sie einen Matrix der Größe 6x2, die Sie als Kartesische Koordinaten interpretieren können ([[x0, y0],[x1, y1],[x2, y2]]). Schreiben Sie eine Funktion, die alle Punkt-Punkt Abstände berechnet und in einer 6x6 Matrix ausgibt.

Abgabe Die Bearbeitungszeit der Teilaufgabe ist für ca. eine Woche ausgelegt, kann aber aufgrund der Anfangshürden etwas Zeitaufwändig wirken. Die Abgabe soll via Moodle bis zu dem dort angegebenen Termin erfolgen. Verspätete Abgaben werden mit einem Abschlag von 3 Punkten je angefangener Woche Verspätung belegt. Geben Sie bitte jeweils nur eine einzige .zip-Datei mit den Quellen Ihrer Lösung ab.

Ich behalte mir vor stichprobenartig Ihre Lösungen in der Übung demonstrieren und erläutern zu lassen. Die Qualität Ihrer Demonstration ist, neben dem abgegebenen Code, ausschlaggebend für die Bewertung!