Python3 Aufgaben

Inhaltsverzeichnis

Author: jz
eMail: jz@mgeg.de
Version: 1.0.0

Vorbemerkungen

Die folgenden Aufgaben orientieren sich an Fragestellungen wie sie in der Praxis so, oder so ähnlich vorkommen.
Bei einigen Aufgaben werden mehrere Fragen gestellt die nach Schwierigkeitsgrad gestaffelt sind. Ziel soll nicht sein alle Fragen richtig zu beantworten. Die schwierigeren Fragen werden für diejenigen Teilnehmer gestellt, denen die Beantwortung der ersten Fragen leichter fallen.
Für die Lösung der Fragen sind viele Lösungswege denkbar. Einige sind eleganter als andere. Entscheidend ist es aber überhaupt ein Lösung zu finden. Bitte beschränken Sie sich bei der Programmierung der Lösungen auf die bisher behandelten Schlüsselwörter/Konstrukte. Diese sind im unteren Teil einer Aufgabe aufgeführt.
Bitte verwenden Sie für die Lösung der Aufgaben nicht Google oder andere Suchmaschinen sondern versuchen Sie selber eine Lösung zu erarbeiten.
Wenn es Probleme gibt die Sie nicht lösen können, untersuchen Sie Ihr Script auf folgende Punkte:

- sind die Einrückungen korrekt (verwenden Sie ausschliesslich vier Leerzeichen zum Einrücken)
- lesen Sie die Fehlermeldung des Interpreters **genau**, die Fehlermeldungen zeigen Ihnen in der Regel die fehlerhafte Zeile sowie die Art des Problems an.
- folgen Sie dem Vier Augen Prinzip; Lassen Sie Ihren Nachbarn über Ihren Code drüber schauen. Häufig erkennt ein Aussenstehender Fehler schneller weil er nicht *betriebsbind* ist.
- Jeden Fehler den Sie selber finden verschafft Ihnen ein Erfolgserlebnis, trotzdem sollten Sie nich zögern um nach Unterstützung zu fragen.
- starten Sie Python3 interaktiv und probieren Sie einzelne Konstrukte in der Python3 Konsole aus

Wer Rechtschreibfehler findet darf sie behalten

Aufgabe 1: Erkennen von IPv4 Adressen

Sie erhalten eine CSV Datei in der IPv4 Adressen eingetragen sind.
Lesen Sie die Datei mit Python3 ein. Das Program soll nun für jede Zeile die unten aufgeführten Fragen beantworten.

1. Handelt es sich bei der aktuellen Zeile um eine IPv4 Host Adresse?
2. Handelt es sich bei der aktuellen Zeile um eine IPv4 Netzmaske?
3. Handelt es sich bei der aktuellen Zeile um eine IPv4 Adresse mit einer Netzmaske in Kurzschreibweise?
4. Handelt es sich bei der aktuellen Zeile um eine IPv4 Broadcast Adresse?
5. Handelt es sich bei der aktuellen Zeile um eine private IPv4 Adresse?
6. Handelt es sich bei der aktuellen Zeile um eine IPv6 Adresse?
7. Handelt es sich bei der aktuellen Zeile um einen gültigen Wert?

Die Aufgaben sind nach Schwierigkeitsgrad gestellt. Schwerere Aufgaben folgen den leichteren Aufgaben.
In der CSV sind mehrere Spalten enthalten. In der ersten Zeile steht eine Beschreibung der einzelnen Spalten.

network_stuff.csv

```
address;private;network;broadcast;netmask;hosts;ipv6;bin
138.201.41.13;;;;;;;
192.168.12.22;;;;;;;
some garbage;;;;;;;
172.32.9.31;;;;;;;
8.8.8.8;;;;;;;
192.168.23.0/24;;;;;;;
13.13.13.15/28;;;;;;;
10.0.12.23/12;;;;;;;
192.168.0.8/29;;;;;;;
7.7.7.7.7;;;;;;;
192.168.23.63/30;;;;;;;
255.255.255.240;;;;;;;
0.0.0.0;;;;;;;
fe80::f486:739b:e9e5:23f6/64;;;;;;;
127.0.1.53;;;;;;;
225.255.255.240/28;;;;;;;
192.168.178.21/24;;;;;;;
192.168.123.255;;;;;;;
```

Schlüsselwörter

- **Kommentare**
- **einfache Variablen**
- **Listen**
- **Dictionaries**
- **print**
- **formatierte Strings** (print(f'{chars}{zeichenkette}{chars}'))
- **if, elif, else**
- **for** oder **while**
- **string.split()**
- **open with**
- **Vergleiche** (< > == >=)
- **Funktionen** (def)
- **Substrings** (string[2:5])
- **input('Wieviel Geld wollen Sie heute ausgeben: ')**
- **list.append(result)**

Aufgabe 1.1: Schreiben einer Funktion

Das obige Programm erkennt zuverlässig IPv4 Adressen. Diese Funktionalität ist für Sie wertvoll. Lagern Sie die obige Funktionalität daher in eine Funktion aus. Die Funktion soll als Argument die IP-Adresse erhalten. Die Funktion soll eine der folgenden Antworten geben:

- öffentliche IPv4 Adresse
- private IPv4 Adresse
- Netzwerkadresse
- Broadcast
- Broadcast
- Netzmaske
- IPv6 Adresse
- etwas Müll

Lösung 1:

Vorbemerkung

In der Realität sind die Dinge häufiger deutlich komplexer als es zunächst den Anschein hat. Bei der Lösung der Aufgabe geht es nicht um eine 100%'tige Lösung, dafür gibt es viel zu viele Sonderfälle zu beachten. In der Praxis würde man zur Lösung dieser Aufgabe auf Bibliotheken zurückgreifen. Je nach Qualität einer Bibliothek sind diese Sonderfälle dann berücksichtigt und der Program Code wäre deutlich einfacher.
Die im weiteren vorgestellte Lösung verwendet die bisher gelernten Schlüsselwörter. Auch hier ist zu beachten das die Lösung keineswegs die beste (geschickteste) Lösung darstellt. Python selber hat eine große Vielzahl von Konstukten um Aufgaben sehr effizient zu lösen. Die Anzahl der Lösungsmöglichkeiten ist schier unendlich, es geht hier um eine einfach nachzuvollziehende Lösung.

Zur Lösung der Aufgabe ist es hilfreich die Aufgabe zunächst in kleinere Aufgaben zu unterteilen. So könnten Sie sich z.B. folgende Liste erstellen um die Aufgabe in kleinere Aufgaben zu zerlegen:

Teillösungen

- lese eine CSV Datei zeilenweise ein
- nehme jede Zeile und extrahiere die erste Spalte
- untersuche die erste Spalte auf ungültige Zeichen und breche den aktuellen Schleifendurchlauf bei ungültigen Zeichen ab
- Die erste Spalte muss aus vier Dezimalzahlen bestehen die durch drei Punkte voneinander getrennt sind.
- Die letzte Zahl kann auch einen / als Trennzeichen zwischen IP und Netzmaske in

Kurzschreibweise enthalten

- Alle Bestandteile können jetzt in Ganze Dezimalzahlen (Integers) umgewandelt werden.
- Sollte die jeweilige Stelle der IP Adresse nicht im Bereich zwischen 0 bis 255 liegen, kann die Bearbeitung der aktuellen Zeile abgebrochen werden.

Bis hierhin ist der erste große Teil der Aufgabe gelöst. Jetzt kommen wir zur Lösung der gegestellten Aufgaben. Im folgenden wird die Aufgabe 1 gelöst, die Lösung der weiteren Fragen erfolgt analog.

eine mögliche Lösung des ersten Teils

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

file = '/home/tn/bin/network_stuff.csv'

first_line = True
# Datei einlesen und die erste Zeile ueberspringen
with open(file, 'r') as fh:
    for line in fh.readlines():
        if first_line is True:
            first_line = False
            continue
        # erste Spalte
        rows = line.split(';')
        pattern = rows[0]
        octett = pattern.split('.')

        # Es kann nur dann eine gültige IP sein, wenn es genau vier Oktetts hat
        lenght = 0
        for i in octett:
            lenght = lenght + 1
        if len != 4:
            continue

        # haben wir eine Netzmaske in Kurzschreibweise
        items = octett[3].split('/')
        if items > 1:
            netmask = items[1]
            ipaddress = [octett[0], octett[1], octett[2], items[0]]
            # naechste Zeile wenn die Netzmaske keine Zahl ist
            try:
                int(netmask)
            except:
                continue
        else:
            netmask = False
            ipaddress = octett

        # enthalten alle otetts der IPadresse nur Integers
```

```
        try:
            for octett in ipaddress:
                int(octett)
        except:
            continue

        # liegen die Integers im Bereich zwischen 0 und 255
        next_line = True
        for octett in ipaddress:
            if octett >= 0 and octett < 256:
                next_line = True
            else:
                next_line = False
                break
        # Abbruch der aktuellen Zeile
        if next_line is False:
            continue

        # jetzt koennen die Fragen nach dem Typ der Adresse beantwortet werden

        # ermitteln der Host Adresse
        # einfachste Annahme, keine Netzmaske
        if netmask is False and octett[3] > 0 and octett[3] < 255:
            print(f'{octett[0]}.{octett[1]}.{octett[2]}.{octett[3]} ist eine Host
Adresse')
        else:
            pass
```

Wann immer Programmcode mehrfach verwendet wird, sollte er in ein Funktion ausgelagert werden

Die erste Aufgabe gilt als gelöst, wenn sich keine Netzmaske ermitteln lässt. Dann kann die IP Adresse als Host Adresse betrachtet werden wenn die erste Stelle der IP Adresse kleiner 255 ist.

In der Realität ist die Frage natürlich etwas komplexer, aber für unsere Zwecke reicht die Lösung aus.

Handelt es sich um eine Host oder um eine Netzadresse

Lösung der Frage ob es sich bei der IP Adresse um eine Netzwerkadresse oder um eine Broadcast Adresse handelt.
Dazu muss eine Netzmaske vorhanden sein. Im folgenden wird jetzt berechnet welche Netzwerkadressen und welche Broadcast Adressen möglich sind. Durch einen Vergleich kann dann ermittelt werden ob es sich um:

- eine Host Adresse handelt
- eine Netzwerk Adresse handelt
- eine Broadcast Adresse handelt

```
print('solution')
```

Aufgabe 2: Versuch und Irrtum

Wenn Sie in Python3 z.B. auf eine Variable zugreifen wollen die nicht existiert bekommen Sie eine Fehlermeldung und die Ausführung des Programmes wird unterbrochen. Es gibt eine Vielzahl von Situationen in denen ein Fehler in der Programmausführung auftritt die zu einem Abbruch des Programmes führen.
Für diese Fälle kennt Python3 das Konzept der Ausnahmebehandlung. Dabei werden die kritischen Code Teile in einen try Block geschrieben. Die Konstruktion sieht wie folgt aus:

nicht zulässige Division durch 0

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

zahl = 12
try:
    ergebnis = zahl / 0
except:
    print('Das war wohl nix, aber es geht weiter')
```

Wenn Sie versuchen eine Zeichenkette in einen Integer (Ganzzahl) umzuwandeln, wird Python mit einer Fehlermeldung abbrechen. Schreiben Sie eine *robuste* Funktion zum umwandeln von Zeichenketten in Integers die als Rückgabe die umgewandelte Zahl oder ein False für den Fehlerfall zurückgibt.

Lösung 2:

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
def get_int(pattern):
    try:
        result = int(pattern)
        return result
    except:
        return False
```

Aufgabe 3: In eine Datei schreiben

In der Aufgaben 1 haben Sie Daten aus einer CSV Datei eingelesen und verarbeitet. Erweitern Sie Ihr Script um die Möglichkeit die ermittelten Daten in eine neue Datei zu schreiben.

Lösung 3:

```
Datei = '/home/tn/bin/einkaufsliste_out.csv'
output = "'Name','Preis','Gewicht','Anzahl'"
with open(Datei, 'a') as fh1:
    fh1.writelines(output)
```

Aufgabe 4: Objekte und Klassen

1. Schreiben Sie ein Klasse mit der Sie neue Produkte anlegen können. Die Eigenschaften des Produktes sollen direkt beim instanzieren eines neuen Objektes übergeben werden.
2. Es soll eine Methode geben mit der sich bestehende Eigenschaften verändern lassen.
3. Um die Daten eines Objektes ausgeben zu können schreiben Sie eine Methode die die Daten zurückgibt.
4. Schreiben Sie eine Methode die die Eigenschaften des Objektes Tabellarisch ausgibt.
5. Schreiben Sie eine Methode die die Eigenschaften des Objektes Tabellarisch ausdruckt wenn Sie das Objekt an print übergeben.

Lösung 4:

lib/Product.py

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

class Product():

    def __init__(self, attribute):
        self.attribute = attribute

    def set_attribute(self, key, value):
        self.attribute[key] = value

    def get_attribute(self):
        return self.attribute

    def show_me(self):
        for _key, _value in self.attribute.items():
            print(f'{_key}: {_value}')

    def __repr__(self):
        output = '-' * 50
        output += '\n'
        for key, value in self.attribute.items():
            output += f'{key}: {value}\n'
        return output
```

Aufgabe 5: Objekte und Klassen

Schreiben Sie eine Klasse in der die Produkt Objekte verwaltet werden können. Die Klasse sollte folgende Methoden enthalten:

1. Einlesen einer CSV Datei mit Produkten
2. anlegen eines neuen Produktes
3. auflisten aller bestehenden Produkte
4. speichern aller Produkte in einer CSV Datei

Lösung 5:

lib/Productlist.py

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

class Productlist():

    def __init__(self, p):
        self.p = p
        self.productlist = []

    def read_csv_file(self, file, delimiter):
        firstline = True
        with open(file, 'r') as fh:
            for line in fh.readlines():
                if firstline is True:
                    firstline = False
                    continue
                items = line.split(delimiter)
                name = items[0][1:]
                name = name[:-1]
                costs = int(items[1])
                weight = int(items[2])
                amount = items[3]
                amount = int(amount[:-1])
                produkt = self.p.Product({'Name': name, 'Preis': costs, 'Gewicht':
weight, 'Anzahl': amount})
                self.productlist.append(produkt)

    def list_products(self):
        for product in self.productlist:
            print(product)

    def add_product(self, product):
        self.productlist.append(product)

    def write_csv_file(self, myfile, delimiter):
        with open(myfile, 'w') as fh:
            line = ''
            for product in self.productlist:
                first_element = True
                for key, value in product.get_attribute().items():
                    value = str(value)
                    if first_element is True:
                        line += f"'{value}'"
                        first_element = False
                    else:
                        line += f"{delimiter}'{value}'"
                line += '\n'
            fh.writelines(line)
```

neue_einkaufsliste.py

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

import lib.Product as p
import lib.Productlist as l

einkaufsliste = l.Productlist(p)
einkaufsliste.read_csv_file('/home/tn/bin/einkaufliste.csv', ',')

while 1:
    name = input('Geben Sie den Produktnamen an: ')
    preis = input('Geben Sie den Preis an: ')
    gewicht = input('Geben Sie das Gewicht an: ')
    anzahl = input('Geben Sie eine Menge an: ')
    product = p.Product({'Name': name, 'Preis': preis, 'Gewicht': gewicht, 'Anzahl':
anzahl})
    einkaufsliste.add_product(product)
    neu = input('Soll ein weiteres Produkt hinzugefuegt werden? (j|n) ')
    if neu != 'j':
        break

einkaufsliste.write_csv_file('/home/tn/bin/einkaufliste-1.csv', ';')
einkaufsliste.list_products()
```

Aufgabe 6: Umwandeln von Formaten

Erweitern Sie das bestehende Script neue_einkaufsliste.py um folgende Funktionalitäten:

1. Schreiben Sie eine Abfrage welche Datei eingelesen werden soll
2. Es soll die Möglichkeit bestehen entweder CSV Dateien oder JSON Dateien einzulesen. Das eingelesene Format soll anhand des Suffixes erkannt werden (.csv, .json)
3. Es soll die Möglichkeit bestehen die Einkaufsliste entweder als CSV oder als JSON -Datei in eine Datei zu schreiben. Die Ausgabe Datei soll das passende Suffix erhalten.

Lösung 6:

lib/Productlist.py

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

import json

class Productlist():
```

```
    def __init__(self, p):
        self.p = p
        self.productlist = []

    def read_csv_file(self, file, delimiter):
        firstline = True
        with open(file, 'r') as fh:
            for line in fh.readlines():
                if firstline is True:
                    firstline = False
                    continue
                items = line.split(delimiter)
                name = items[0][1:]
                name = name[:-1]
                costs = int(items[1])
                weight = int(items[2])
                amount = items[3]
                amount = int(amount[:-1])
                product = self.p.Product({'Name': name, 'Preis': costs, 'Gewicht':
weight, 'Anzahl': amount})
                self.productlist.append(product)

    def read_json_file(self, file):
        with open(file, 'r') as fh:
            line = fh.readline()
            data = json.loads(line)
            for product in data:
                _product = self.p.Product(product)
                self.productlist.append(_product)

    def list_products(self):
        for product in self.productlist:
            print(product)

    def add_product(self, product):
        self.productlist.append(product)

    def write_csv_file(self, myfile, delimiter):
        with open(myfile, 'w') as fh:
            line = ''
            for product in self.productlist:
                first_element = True
                for key, value in product.get_attribute().items():
                    value = str(value)
                    if first_element is True:
                        line += f"{value}"
                        first_element = False
                    else:
                        line += f"{delimiter}{value}"
                line += '\n'
            fh.writelines(line)
```

```
    def write_json_file(self, myfile):
        with open(myfile, 'w') as fh:
            products = []
            for product in self.productlist:
                _product = {}
                for key, value in product.get_attribute().items():
                    _product[key] = value
                products.append(_product)
            fh.writelines(json.dumps(products))
```

neue_einkaufsliste.py

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

import os
import time
import lib.Product as p
import lib.Productlist as l

os.system('clear')
einkaufsliste = l.Productlist(p)

print(50 * '-')
input_file = input('Geben Sie die Datei an die eingelesen werden soll: ')
print(input_file[-3:])
print(input_file[-4:])
if input_file[-3:] == 'csv':
    delimiter = input('Geben Sie das Trennzeichen der CSV Datei an: ')
    einkaufsliste.read_csv_file(input_file, delimiter)
elif input_file[-4:] == 'json':
    einkaufsliste.read_json_file(input_file)
else:
    print('Die Datei hat kein gültiges Suffix!')
    exit(1)

while 1:
    print('Wählen Sie ein Zahl für eine Aktion aus')
    print('1 - neues Produkt anlegen')
    print('2 - bestehende Daten in eine CSV Datei schreiben')
    print('3 - bestehende Daten in eine JSON Datei schreiben')
    print('4 - alle Produkte auflisten')
    print('5 - Program beenden')
    cmd = input('Wählen Sie ein Zahl für eine Aktion aus: ')

    if cmd == '1':
        name = input('Geben Sie den Produktnamen an: ')
        preis = input('Geben Sie den Preis an: ')
        gewicht = input('Geben Sie das Gewicht an: ')
        anzahl = input('Geben Sie eine Menge an: ')
```

```
        product = p.Product({'Name': name, 'Preis': preis, 'Gewicht': gewicht,
'Anzahl': anzahl})
        einkaufsliste.add_product(product)
    elif cmd == '2':
        output_file = input('Geben Sie die Datei an die geschrieben werden soll: ')
        delimiter = input('Geben Sie das Trennzeichen der CSV Datei an: ')
        einkaufsliste.write_csv_file(output_file, delimiter)
    elif cmd == '3':
        output_file = input('Geben Sie die Datei an die geschrieben werden soll: ')
        einkaufsliste.write_json_file(output_file)
    elif cmd == '4':
        einkaufsliste.list_products()
    elif cmd == '5':
        exit(0)
    else:
        print('Ihre Eingabe war ungültig')
        time.sleep(10)
```