27 CSV-Dateien

27.1 Beschreibung

- CSV ist ein Format für Textdateien und steht für Comma-Separated-Values
- mit diesen Textdateien lassen sich strukturierte Daten speichern, meistens in Tabellenform
- diese Textdateien dienen als Austauschformat zwischen verschiedenen Anwendungen und Datenbanken
- es wird die Dateiendung csv verwendet
- es gibt keinen allgemein gültigen Standard
- prinzipieller Aufbau einer CSV-Datei
    - zwischen den einzelnen Datenfelder steht ein Trennzeichen – meistens das Komma
    - bei Gleitpunktzahlen wird der Punkt verwendet
    - für jede Tabellenzeile gibt es eine neue Zeile in der CSV-Datei
    - in der ersten Zeile werden häufig die Feldbezeichnungen angeben

- Beispiel

```
Nr,Vorname,Nachname,Straße,Email,Eintrittsdatum
1,Marli,Moor,Giescheider Weg 82,marlimoor@justmail.none,03.08.2017
2,Joschka,Klingel,Diedenberg 13,joschka15@xyz.none,17.12.2014
3,Egidius,Springmann,Karl-Str 99,es.1991@domain.none,13.12.1995
```

27.2 Dateizugriff auf CSV-Datei - Basis

27.2.1 Dateizugriff auf CSV-Datei – Basis - lesend

- Dateizugriff (lesend auf eine Textdatei)
- liefert für jede Zeile der CSV-Datei eine Zeichenkette (string)

```
# CSV-Datei öffnen-
d_csv = open("csv_datei.csv", "r", encoding="UTF-8")

# CSV-Datei zeilenweise lesen
# die einzelnen Werte werden z.B. mit split ermittelt bzw. separiert
# .replace ersetzt den Zeilenvorschub durch einen leeren String
# .split teilt an den Trennzeichen auf
for zeile in d_csv:
    print(zeile, end="")
    print(type(zeile))
    print(zeile.replace("\n", "").split(","))

# CSV-Datei schließen
d_csv.close()
```

- beim Einlesen einer Dateizeile erhält man eine Zeichenkette (string)
- in der Zeichenkette stehen die einzelnen Werte hintereinander
- zwischen den einzelnen Werten in einer Zeile stehen die Trennzeichen
- die einzelnen Werte sind durch eigene Python-Befehle zu ermitteln

- Beispielausgabe:

```
Nr,Vorname,Nachname,Strasse,Email,Datum
<class 'str'>
['Nr', 'Vorname', 'Nachname', 'Strasse', 'Email', 'Datum']

1,Marli,Moor,Giescheider Weg 82,marlimoor@justmail.none,03.08.2017
<class 'str'>
['1', 'Marli', 'Moor', 'Giescheider Weg 82', 'marlimoor@justmail.none',
'03.08.2017']

2,Joschka,Klingel,Diedenberg 13,joschka15@xyz.none,17.12.2014
<class 'str'>
['2', 'Joschka', 'Klingel', 'Diedenberg 13', 'joschka15@xyz.none',
'17.12.2014']

3,Egidius,Springmann,Karl-Str 99,es.1991@domain.none,13.12.1995
<class 'str'>
['3', 'Egidius', 'Springmann', 'Karl-Str 99', 'es.1991@domain.none',
'13.12.1995']
```

27.2.2 Dateizugriff auf CSV-Datei – Basis - schreibend

- Dateizugriff (schreibend auf eine Textdatei)
- liefert für jede Zeile der CSV-Datei eine Zeichenkette (string)

```
# CSV-Datei öffnen-
d_csv = open("csv_ausgabe.csv", "r", encoding="UTF-8")

# CSV-Datei zum Schreiben öffnen
try:
   d_csv = open("csv_ausgabe.csv", "w", encoding="UTF-8")
except:
   print("Zugriffsfehler auf die CSV-Datei")

# CSV-Datei zeilenweise schreiben
# eine Liste von Zeichenketten enthält die Spaltenüberschriften gefolgt von den Datenzeilen
# zwischen den Spaltenüberschriften und den einzelnen Werten sind Trennzeichen
# jeder Datenzeile ist ein Zeilenvorschub \n hinzuzufügen
# die Liste von Zeichenketten wird mit writelines in die Datei geschrieben
zeilen = ["Nr,Vorname,Nachname,Strasse,Email,Datum"+"\n",
        "1,Marli,Moor,Giescheider Weg 82," \
        "marlimoor@justmail.none,03.08.2017"+"\n",
        "2,Joschka,Klingel,Diedenberg 13," \
        "joschka15@xyz.none,17.12.2014"+"\n",
        "3,Egidius,Springmann,Karl-Str 99," \
        "es.1991@domain.none,13.12.1995"+"\n"]

# Liste der Datenzeilen in CSV-Datei zeilenweise schreiben
d_csv.writelines(zeilen)

# CSV-Datei schließen
d_csv.close()
```

- Beispiel: Datei csv_ausgabe.csv

```
Nr,Vorname,Nachname,Strasse,Email,Datum
1,Marli,Moor,Giescheider Weg 82,marlimoor@justmail.none,03.08.2017
2,Joschka,Klingel,Diedenberg 13,joschka15@xyz.none,17.12.2014
3,Egidius,Springmann,Karl-Str 99,es.1991@domain.none,13.12.1995
```

27.3 Dateizugriff auf CSV-Datei – CSV-Reader

27.3.1 Dateizugriff auf CSV-Datei – CSV-Reader – lesend

- Dateizugriff, lesend, mit CSV-Reader (Modul csv)
- separiert die Einzelwerte und liefert eine Liste (list)
- Einzelwerte einer Zeile lassen sich mit list[0], list[1] usw. ansprechen

```
# Modul csv einbinden
import csv

# CSV-Datei öffnen-
d_csv = open("csv_datei.csv", "r", encoding="UTF-8")

# CSV-Datei zeilenweise lesen -
# reader-Objekt zum "CSV-Lesen" der CSV-Datei erstellen
csv_reader = csv.reader(d_csv)

# CSV-Datei zeilenweise lesen
# es wird eine Liste mit den einzelnen Werten in einer Zeile
bereitgestellt
for zeile in csv_reader:
    print(zeile)
    print(type(zeile))

# CSV-Datei schließen
d_csv.close()
```

- beim Einlesen einer Zeile mit dem CSV-Reader wird die Zeile bei den Trennzeichen aufgeteilt
- die einzelnen Werte einer werden als Liste von Werten bereitgestellt

- Beispielausgabe:

```
['Nr', 'Vorname', 'Nachname', 'Strasse', 'Email', 'Datum']
<class 'list'>

['1', 'Marli', 'Moor', 'Giescheider Weg 82', 'marlimoor@justmail.none',
'03.08.2017']
<class 'list'>

['2', 'Joschka', 'Klingel', 'Diedenberg 13', 'joschka15@xyz.none',
'17.12.2014']
<class 'list'>

['3', 'Egidius', 'Springmann', 'Karl-Str 99', 'es.1991@domain.none',
'13.12.1995']
<class 'list'>
```

27.3.2 Dateizugriff auf CSV-Datei – CSV-Writer – schreibend

- Dateizugriff, schreibend, mit CSV-Writer (Modul csv)
- Liste mit den Spaltenüberschriften (Spaltenkopf) wird als erste Zeile mit **.writerow**() ausgegeben
- Liste der Datenzeilen mit **.writerows**() ausgegeben

```
# Modul csv einbinden
import csv

# CSV-Datei zum Schreiben öffnen und CSV-Writer-Objekt erstellen
try:
    d_csv = open("csv_ausgabe.csv", "w", encoding="UTF-8", newline="")
    csv_writer = csv.writer(d_csv)
except:
    print("Zugriffsfehler auf die CSV-Datei")

# CSV-Datei zeilenweise schreiben
# es wird eine Liste mit den einzelnen Werten in einer Zeile
bereitgestellt
daten = [["1","Marli","Moor","Giescheider Weg 82",
          "marlimoor@justmail.none","03.08.2017"],
         ["2","Joschka","Klingel","Diedenberg 13",
          "joschka15@xyz.none","17.12.2014"],
         ["3","Egidius","Springmann","Karl-Str 99",
          "es.1991@domain.none","13.12.1995"]]

# Spaltenköpfen in CSV-Datei ausgegeben
csv_writer.writerow(["Nr","Vorname","Nachname","Strasse",
                     "Email","Datum"])

# Liste der Daten zeilenweise in die CSV-Datei schreiben
csv_writer.writerows(daten)

# CSV-Datei schließen
d_csv.close()
```

- Beispiel: Datei csv_ausgabe.csv

```
Nr,Vorname,Nachname,Strasse,Email,Datum
1,Marli,Moor,Giescheider Weg 82,marlimoor@justmail.none,03.08.2017
2,Joschka,Klingel,Diedenberg 13,joschka15@xyz.none,17.12.2014
3,Egidius,Springmann,Karl-Str 99,es.1991@domain.none,13.12.1995
```

27.4 Dateizugriff auf CSV-Datei – CSV-DictReader

27.4.1 Dateizugriff auf CSV-Datei – CSV-DictReader – lesend

- Dateizugriff, lesend, mit CSV-DictReader (Modul csv)
- in der ersten Zeile stehen die Spaltennamen
- separiert die Einzelwerte und liefert ein assoziatives Array (dictionary) pro Zeile
- Einzelwerte einer Zeile lassen sich mit etwa mit dict["**name**"], dict["**datum**"] usw. ansprechen

```
# Modul csv einbinden
import csv

try:
    d_csv = open("csv_datei.csv", "r", encoding="UTF-8")
except:
    print("Zugriffsfehler auf die CSV-Datei")

# CSV-Dict-Reader erstellen
csv_reader = csv.DictReader(d_csv)

# CSV-Datei zeilenweise lesen
# es wird ein assoziatives Array (dictionary) mit den einzelnen Werten
in einer Zeile bereitgestellt
# die Einzelwerte lassen sich mit dict["name"] ansprechen
for zeile in csv_reader:
    print(zeile)
    print(type(zeile))

    for i in zeile.keys():
        print(f"{i:10}: {zeile[i]}")

# CSV-Datei schließen
d_csv.close()
```

- beim Einlesen einer Zeile mit dem CSV-DictReader wird die Zeile bei den Trennzeichen aufgeteilt und
- die einzelnen Werte einer werden den jeweiligen Spaltenname aus der ersten Zeile zugeordnet

27.4.2 Dateizugriff auf CSV-Datei – CSV-DictWriter – schreibend

- Dateizugriff, schreibend, mit CSV-DictWriter (Modul csv)
- Liste mit den Spaltenüberschriften (Spaltenkopf) wird als erste Zeile mit **.writeheader**() ausgegeben
- Liste der Datenzeilen mit **.writerows**() ausgegeben

- in der ersten Zeile stehen die Spaltennamen
- ansprechen

```
# Modul csv einbinden
import csv

# CSV-Datei zum Schreiben öffnen und CSV-DictWriter-Objekt erstellen
try:
    d_csv = open("csv_ausgabe.csv", "w", encoding="UTF-8", newline="")
except:
    print("Zugriffsfehler auf die CSV-Datei")

# CSV-Dict-Writer erstellen
# es wird eine Liste mit den Spaltenköpfen ausgegeben
spaltenkopf = ["Nr","Vorname","Nachname","Strasse","Email","Datum"]
csv_writer = csv.DictWriter(d_csv, spaltenkopf)
csv_writer.writeheader()

# CSV-Datei zeilenweise schreiben
# es wird eine Liste von mit den einzelnen Werten in einer Zeile
bereitgestellt
daten = [{"Nr":"1", "Vorname":"Marli", "Nachname":"Moor",
          "Strasse":"Giescheider Weg 82",
"Email":"marlimoor@justmail.none",
          "Datum":"03.08.2017"},
         {"Nr": "2", "Vorname": "Joschka", "Nachname": "Klingel",
          "Strasse": "Diedenberg 13", "Email": "joschka15@xyz.none",
          "Datum": "17.12.2014"},
         {"Nr": "3", "Vorname": "Egidius", "Nachname": "Springmann",
          "Strasse": "Karl-Str 99", "Email": "es.1991@domain.none",
          "Datum": "13.12.1995"}]

# CSV-Datei zeilenweise schreiben
# die Liste der Zeilen werden in die Datei geschrieben
csv_writer.writerows(daten)

# CSV-Datei schließen
d_csv.close()
```

- Beispiel: Datei csv_ausgabe.csv

```
Nr,Vorname,Nachname,Strasse,Email,Datum
1,Marli,Moor,Giescheider Weg 82,marlimoor@justmail.none,03.08.2017
2,Joschka,Klingel,Diedenberg 13,joschka15@xyz.none,17.12.2014
3,Egidius,Springmann,Karl-Str 99,es.1991@domain.none,13.12.1995
```

Aufgabe

1. In einer CSV-Datei ist eine Adressübersicht abzuspeichern.
   In dieser Datei sind Name, Vorname, Strasse, PLZ, Wohnort, Geburtsdatum und die Telefonnr von 5 Personen abzuspeichern.

2. Diese CSV-Datei ist über den Basiszugriff auf eine Textdatei einzulesen und die Einträge formatiert auf der Konsole auszugeben.

3. Diese CSV-Datei ist über ein reader-Objekt aus dem csv-Modul einzulesen und die Einträge formatiert auf der Konsole auszugeben.

4. Diese CSV-Datei ist über ein readerDict-Objekt aus dem csv-Modul einzulesen und die Einträge formatiert auf der Konsole auszugeben.

5. Die CSV-Datei ist um die Daten von zwei weiteren Personen zu ergänzen und über eines der Lesemodule (siehe Aufgabe 2 oder Aufgabe 3 oder Aufgabe 4) zu protokollieren