Präsenzübungen: Datenverarbeitung mit Python

Jowan Sulaiman
ABS IT

3. Juni 2025

Präsenzübungen 1: Python File Handling 🗁

Übung 1.1: "Hallo, Datei!"(Schreiben & Lesen)

`Ziel:` Grundlagen des Schreibens in eine Datei und des Lesens aus einer Datei verstehen.

`Aufgaben:`

1. Erstellt ein Python-Skript.
2. Lasst das Skript euren Vornamen in eine neue Datei namens `mein_name.txt` schreiben.
3. Stellt sicher, dass die Datei nach dem Schreiben geschlossen wird (oder verwendet das `with`-Statement).
4. Erweitert das Skript: Öffnet `mein_name.txt` erneut, lest den Inhalt aus und gebt ihn auf der Konsole aus.

`Diskussionspunkte:`

- Welche Modi habt ihr für `open()` verwendet? Warum?
- Was passiert, wenn das Skript mehrmals ausgeführt wird (besonders im Schreibmodus `"w"`)?
- Wie stellt ihr sicher, dass die Datei geschlossen wird?

Übung 1.2: "Liederliste"(Zeilenweise lesen und anhängen)

`Ziel:` Zeilenweises Verarbeiten und Anhängen von Inhalten an eine Datei.

`Vorbereitung:` Erstellt manuell eine Datei `lieder.txt` mit 2-3 eurer Lieblingslieder, jedes Lied in einer neuen Zeile.

`Aufgaben:`

1. Schreibt ein Python-Skript, das den Benutzer nach einem weiteren Lieblingslied fragt.
2. Dieses neue Lied soll an die Datei `lieder.txt` angehängt werden (in einer neuen Zeile).
3. Anschließend soll das Skript die gesamte Datei `lieder.txt` Zeile für Zeile lesen und jedes Lied mit einer vorangestellten Nummer auf der Konsole ausgeben (z.B. "1. Yesterday", "2. Bohemian Rhapsody", ...).

`Diskussionspunkte:`

- Welchen Modus braucht ihr zum Anhängen?
- Wie könnt ihr die Zeilennummerierung beim Ausgeben realisieren?
- Was ist der Vorteil, die Datei Zeile für Zeile zu lesen, wenn sie sehr lang wäre?

Übung 1.3: "Wort Zähler"(Datei lesen und einfache Textanalyse)

`Ziel:` Inhalt einer Datei lesen und einfache Operationen auf dem Text durchführen.

`Vorbereitung:` Erstellt eine Datei `kurzgeschichte.txt` mit einem kurzen Text (3-4 Sätze).

`Aufgaben:`

1. Schreibt ein Python-Skript, das `kurzgeschichte.txt` öffnet und den gesamten Inhalt einliest.

2. Zählt, wie oft ein bestimmtes Wort (z.B. ündöder ein anderes häufiges Wort) im Text vorkommt. Ignoriert dabei Groß- und Kleinschreibung.
3. Gebt das Wort und seine Häufigkeit aus.

`Diskussionspunkte:`

- Wie könnt ihr den Text am besten nach Wörtern aufteilen? (`split()`)
- Wie stellt ihr sicher, dass Groß- und Kleinschreibung ignoriert wird? (`lower()` oder `upper()`)

Präsenzübungen 2: JSON in Python

Übung 2.1: "Meine Visitenkarte als JSON"(Python-Dict zu JSON)

`Ziel:` Ein Python-Dictionary erstellen und als JSON-String sowie in eine Datei serialisieren.

`Aufgaben:`

1. Erstellt ein Python-Dictionary, das eure "VisitenkartenDaten enthält: `name`, `beruf`, `email`, `telefonnummer` (als Strings).
2. Verwendet `json.dumps()`, um dieses Dictionary in einen JSON-String umzuwandeln. Gebt diesen String auf der Konsole aus. Verwendet den `indent`-Parameter für eine schöne Formatierung.
3. Speichert dasselbe Dictionary nun mit `json.dump()` in eine Datei namens `visitenkarte.json`. Auch hier auf eine lesbare Formatierung achten.

`Diskussionspunkte:`

- Wie sieht der JSON-String im Vergleich zum Python-Dictionary aus?
- Was macht `indent` genau?
- Was ist der Unterschied zwischen `dumps` und `dump`?

Übung 2.2: "Visitenkarte laden"(JSON-Datei zu Python-Dict)

`Ziel:` Eine JSON-Datei deserialisieren und auf die Daten zugreifen.

`Vorbereitung:` Verwendet die in Übung 2.1 erstellte `visitenkarte.json`.

`Aufgaben:`

1. Schreibt ein Python-Skript, das `visitenkarte.json` öffnet und den Inhalt mit `json.load()` in ein Python-Dictionary lädt.
2. Greift auf die einzelnen Werte im Dictionary zu (Name, Beruf etc.) und gebt sie formatiert auf der Konsole aus (z.B. "Name: Max Mustermann").

`Diskussionspunkte:`

- Welchen Datentyp hat das Objekt nach dem Laden mit `json.load()`?
- Wie greift man auf die Werte in einem Dictionary zu?
- Was würde passieren, wenn ein Schlüssel nicht im JSON vorhanden wäre? (ggf. `get()`-Methode diskutieren)

Übung 2.3: SSchnelle Konfig"(JSON-String direkt parsen)

`Ziel:` Einen JSON-formatierten String direkt im Code parsen.

`Aufgaben:`

1. Definiert einen Python-String, der eine einfache JSON-Struktur enthält, z.B.:

```
config_string = '{"theme": "dark", "fontSize": 12, "notifications": true}'
```

2. Verwendet `json.loads()`, um diesen String in ein Python-Dictionary umzuwandeln.
3. Gebt das `theme` und die `fontSize` aus dem resultierenden Dictionary aus.

`Diskussionspunkte:`

- Warum ist `json.loads()` hier die richtige Wahl und nicht `json.load()`?
- Auf welche Fehler könnte man stoßen, wenn der String kein valides JSON ist?

Präsenzübungen 3: RegEx in Python

Übung 3.1: "Finde alle Zahlen"(`re.findall`)

`Ziel:` Das Grundkonzept von `re.findall` und einfachen Mustern verstehen.

`Aufgaben:`

1. Gegeben sei der String: `text = Ïch habe 3 Äpfel und 15 Birnen für 4 Personen."`
2. Schreibt ein Python-Skript mit dem `re`-Modul, das alle Zahlen aus diesem Satz extrahiert und als Liste ausgibt.

`Diskussionspunkte:`

- Welches RegEx-Muster habt ihr verwendet, um Zahlen zu finden? (`\d+`?)
- Was ist der Datentyp des Ergebnisses von `re.findall()`?

Tipp

Das Muster `r"\d+"` sucht nach einer oder mehreren Ziffern.

Übung 3.2: "Beginnt mit..."(Einfaches `re.match` oder `re.search` mit `^`)

`Ziel:` `re.match()` oder `re.search()` für eine einfache Musterprüfung am Stringanfang verwenden.

`Aufgaben:`

1. Lasst den Benutzer einen Satz eingeben.
2. Schreibt ein Python-Skript, das prüft, ob der eingegebene Satz mit dem Wort `"Das"` (genau diese Schreibweise) beginnt. Gebt "Jaöder "Neinäus.

`Diskussionspunkte:`

- Welchen Unterschied gibt es zwischen `re.match()` und `re.search()` mit dem Anker `^`?
- Wie würde das Muster aussehen, wenn auch "das"(kleingeschrieben) am Anfang erlaubt wäre?

Übung 3.3: Ërsetze Leerzeichen"(Einfaches `re.sub`)

`Ziel:` Die Grundfunktionalität von `re.sub` verstehen.

`Aufgaben:`

1. Gegeben sei der Satz: `text = "Python ist wirklich vielseitig."`
2. Schreibt ein Python-Skript, das alle Vorkommen von einem oder mehreren Leerzeichen durch ein einzelnes Leerzeichen ersetzt und den korrigierten Satz ausgibt.

`Diskussionspunkte:`

- Welches Muster habt ihr für ëin oder mehrere Leerzeichen"verwendet? (`\s+`?)
- Gibt es andere Zeichen, die `\s` auch erfasst (z.B. Tabs, Zeilenumbrüche)?

Tipp

Das Muster `r"\s+"` sucht nach einem oder mehreren Whitespace-Zeichen.