Danny Rehl, Programmieren I - Python

Übungsblatt 02

Abgabefrist: Montag, den 04.11.2024, um 23:59 Uhr

Wichtig: Sie finden für dieses Übungsblatt eine Vorlage für Ihre Abgabe zum Herunterladen im Moodle.

Aufgabe 1 (1½ Punkte)

Weisen Sie den Variablen `A` bis `D` passende Integer-Werte zu, sodass die folgenden Ausgaben entstehen:

```
>>> print(A + 2 * B)
35671
>>> print((3 ** (C - 1)) + 1)
19684
>>> print(True and not D - 37)
True
```

Hinweis: Beachten Sie Operator-Präzedenz. Also zum Beispiel:

```
>>> 1 + 2 * 3
7
>>> (1 + 2) * 3
9
```

Aufgabe 2 (2 Punkte)

Das folgende Zitat aus dem Wikipedia-Artikel zu Python ist in der Variablen `QUOTE` gespeichert:

> Python ist eine universelle, üblicherweise interpretierte höhere Programmiersprache.

Weisen Sie den Variablen `F` bis `N` passende Integer-Werte zu, wobei `I`, `K` und `L` *negative* Integer-Werte sein sollen, sodass nach Anwenden von Indexing und Slicing auf den String folgende Ausgaben entstehen:

```
>>> print(QUOTE[F:G] + QUOTE[H] + QUOTE[I] + 2 * QUOTE[J] + QUOTE[K])
Python ist toll.
>>> print(QUOTE[:L] + "su" + QUOTE[48:M:2] + QUOTE[-6::N])
Python ist super.
```

Aufgabe 3 (2 Punkte)

Wir wollen in den folgenden Aufgaben eine erste rudimentäre Korpusanalyse durchführen.

Die Variable `CORPUS_TEXT` enthält einen Beispieltext, welcher nicht verändert werden soll. In den folgenden Aufgaben werden verschiedene Operationen auf dem enthaltenen Text durchgeführt; die Ergebnisse dieser Operationen werden in anderen Variablen gespeichert werden, sodass `CORPUS_TEXT` unverändert bleibt.

Wir möchten zunächst herausfinden, wie viele Sätze das Korpus enthält. Wir definieren einen **Satz** vereinfacht als einen String, der mit einem Punkt `.` endet. Sie können davon ausgehen, dass keine anderen Satzzeichen wie `!` oder `?` ein Satzende markieren.

Als ersten Schritt soll `CORPUS_TEXT` in seine Sätze aufgeteilt und das Ergebnis in `SENTENCES` gespeichert werden. Die Anzahl der Elemente in `SENTENCES` entspricht also der Anzahl der Sätze im Korpus. Diese Zahl soll in `SENTENCES_COUNT` gespeichert werden. Dabei ist `SENTENCES` eine Liste von Typ `list` und `SENTENCES_COUNT` eine Ganzzahl vom Typ `int`.

Es gibt mehrere korrekte Varianten für `SENTENCES`, denn zwischen den Sätzen befindet sich ein Punkt und ein Leerzeichen. Je nachdem, was man genau noch als Bestandteil eines Satzes definiert, erhält man unterschiedliche Ergebnisse für die Elemente in `SENTENCES`. Das spielt für diese Aufgabe allerdings keine Rolle, denn für die gesuchte *Anzahl* der Sätze ist diese Information irrelevant.

Tipp: Zum Trennen eines Strings in mehrere Substrings gibt es eine passende String-Methode. Zum Zählen der Elemente einer Liste gibt es eine passende Built-in-Funktion. Ganz allgemein werden sich die Methoden von Variablen vom Typ String für die folgenden Aufgaben als äußerst nützlich erweisen. Es lohnt sich also, alle einfach mal anzuschauen, um einen Überblick zu bekommen, falls man nicht direkt eine Lösungsidee findet. Außerdem kann es sinnvoll sein, mehrere `print`-Befehle zum "Debugging" einzubauen, also die Werte der verwendeten Variablen in Zwischenschritten anzuschauen.

Häufiger Fehler: Bedenken Sie, dass auch der letzte Satz mit einem Punkt endet und was das eventuell für den letzten Eintrag in Ihrer Liste bedeutet, wenn Sie `str.split` verwenden. Sie sollten sich im Zweifel Ihr Zwischenergebnis mit `print(SENTENCES)` anschauen.

Wichtig: Ihr Code muss allgemein formuliert sein und prinzipiell mit jedem beliebigen Text funktionieren, nicht nur mit dem gegebenen Beispieltext.

Aufgabe 4 (2½ Punkte)

Wir möchten die Anzahl der Token und Types im Korpus bestimmen. Dazu folgende Definitionen:

- Ein **Token** definieren wir als eine Reihe von Zeichen, die zwischen zwei Leerzeichen oder Satzzeichen beziehungsweise am Anfang des Strings steht. Satzzeichen sind für dieses Aufgabenblatt keine Tokens. Zum Beispiel enthält der String `'Das Pferd ist schwarz.'` insgesamt 4 Tokens: `'Das'`, `'Pferd'`, `'ist'` und `'schwarz'`.
- Unter **Tokenisierung** verstehen wir die Aufteilung eines Textes in seine einzelnen Tokens.
- Einen **Type** definieren wir als ein Wort, welches in seiner Form zwar mehrfach vorkommen darf, aber nur einmal gezählt wird. Bei unserer Definition soll Groß- und Kleinschreibung *nicht* vernachlässigt werden. Beispielsweise enthält der String `'Dieses Auto ist schwarz und dieses Auto ist rot.'` insgesamt 9 Tokens und davon wiederum 7 Types: `'Dieses'`, `'Auto'`, `'ist'`, `'schwarz'`, `'und'`, `'dieses'` und `'rot'`. Die beiden Types `'Auto'` und `'ist'` kommen jeweils 2 Mal vor. Die beiden Token `'Dieses'` und `'dieses'` gelten wegen der unterschiedlichen Groß- und Kleinschreibung als unterschiedliche Types, welche jeweils 1 Mal vorkommen.

Dabei stören zunächst die Satzzeichen, welche laut unserer Definition kein Token sind. Sie können davon ausgehen, dass in den Texten nur Punkt, Doppelpunkt und Komma als Satzzeichen vorkommen (kein Semikolon oder ähnliches) und auch jeder Punkt, Doppelpunkt oder Komma ein Satzzeichen ist (keine Kommas in einer Dezimalzahl oder ähnliches). Entfernen Sie alle Satzzeichen im Text von `CORPUS_TEXT` und speichern Sie das Ergebnis als `CORPUS_CLEANED`. Die Variable `CORPUS_CLEANED` soll ein String vom Typ `str` sein.

Tokenisieren Sie das Korpus in `CORPUS_CLEANED` und weisen Sie das Ergebnis der Variablen `TOKENS` zu. Anschließend soll in `TOKENS_COUNT` die Anzahl der Tokens gespeichert werden. Dabei soll `TOKENS` eine Liste vom Typ `list` sein und `TOKENS_COUNT` eine Ganzzahl vom Typ `int`.

Bestimmen Sie nun die Types sowie deren Anzahl und speichern Sie das Ergebnis in den Variablen `TYPES` beziehungsweise `TYPES_COUNT` ab. Dabei ist `TYPES` eine aufzählbare Variable und `TYPES_COUNT` eine Ganzzahl vom Typ `int`.

Tipp: Der Datentyp von `TYPES` ist absichtlich so frei gehalten. Sie haben einen Datentyp kennengelernt, welcher hierfür besonders geeignet ist. Aber auch andere aufzählbare Datentypen wie `list` sind erlaubt.

Aufgabe 5 (2 Punkte)

Das Korpus soll bezüglich der in der Variablen `KEYWORDS` gespeicherten Schlüsselwörter untersucht werden. Dabei ist `KEYWORDS` - genauso wie schon `CORPUS_TEXT` - nur ein Beispiel und Ihr Code soll unabhängig von den Beispieleinträgen funktionieren.

Erstellen Sie eine Variable `FEATURES` vom Typ `dict`, welche als Schlüssel die Elemente aus `KEYWORDS` enthält. Die Werte der Schlüssel-Wert-Paare sollen jeweils eine Liste mit folgenden Informationen über das jeweilige Schlüsselwort sein:

1. Die Länge des Schlüsselworts. Der Wert soll vom Typ Integer `int` sein.
2. Ob das Schlüsselwort eine Ganzzahl ist. Der Wert soll vom Typ Boolean `bool` sein. (Tipp: Es gibt eine passende String-Methode.)
3. Ob das Schlüsselwort mit einem Großbuchstaben beginnt. Der Wert soll vom Typ Boolean sein. (Tipp: Auch hier gibt es wieder eine passende String-Methode in Verbindung mit Indexing/Slicing.)
4. Die Anzahl der Vorkommen des Schlüsselworts in `CORPUS_TEXT`. Es ist also der Text als String gemeint, ohne Tokenisierung. Der Wert soll vom Typ Integer sein. (Tipp: Es gibt eine passende String-Methode.)

Beispiel:

```
CORPUS_TEXT = 'Python reached version 1 in January 1994.'
KEYWORDS = ['Python', '1']
```

Erwartetes Ergebnis für das Beispiel:

```
>>> FEATURES
{'Python': [6, False, True, 1], '1': [1, True, False, 2]}
```

Tipp: Da die Anzahl der Schlüsselwörter variabel ist, müssen Sie eine passende Kontrollstruktur verwenden, um über die Einträge zu iterieren. Es bietet sich beispielsweise eine `for`-Schleife an.

Bemerkung: Machen Sie sich beim jeweils letzten Wert klar, warum hier andere Ergebnisse entstehen können, wenn Sie die Vorkommen nicht in im String `CORPUS_TEXT`, sondern in der Liste `TOKENS` gezählt hätten.

Zusatzaufgabe 6 (keine Punkte)

Wir möchten den fünften Buchstaben des Worts “Hallo” ausgeben. Dazu weisen wir das Wort als String der Variablen `S` mittels `S = 'Hallo'` zu. Da “Hallo” fünf Buchstaben hat, schreiben wir `print(S[5])` zum Ausgeben des fünften Buchstabens. Statt des gewünschten Buchstabens “o” erhalten wir jedoch folgenden Fehler:

```
>>> S = 'Hallo'
>>> print(S[5])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

Was haben wir falsch gemacht? Es gibt eine Eigenschaft von Sequences in Python, welche dafür verantwortlich ist.

Sie können Ihre Ideen hierzu als Kommentar in der Abgabedatei notieren. Es wird allerdings keine automatische Rückmeldung von Tutron geben, sondern Sie können die Aufgabe bei Bedarf gemeinsam im Tutorium diskutieren.

Zusatzaufgabe 7 (keine Punkte)

Wir möchten jeweils den letzten Eintrag in unseren beiden Variablen `T = [1, 2, 3]` und `U = (1, 2, 3)` von `3` auf `4` ändern. Bei `T` funktioniert das auch, bei `U` erscheint allerdings folgender Fehler:

```
>>> T = [1, 2, 3]
>>> U = (1, 2, 3)
>>> print(T[-1])
3
>>> T[-1] = 4
>>> print(T[-1])
4
>>> print(U[-1])
3
>>> U[-1] = 4
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

Anscheinend können wir bei Variablen vom Typ `tuple` keine Einträge ändern. Was ist der Grund hierfür? (Es gibt einen speziellen Begriff für diese Eigenschaft von bestimmten Datentypen in Python.)

Sie können Ihre Ideen hierzu als Kommentar in der Abgabedatei notieren. Es wird allerdings keine automatische Rückmeldung von Tutron geben, sondern Sie können die Aufgabe bei Bedarf gemeinsam im Tutorium diskutieren.