Übung 4: Dictionaries und reguläre Ausdrücke

Bill Bosshard 12-933-255
Lukas Vollenweider 13-751-888

1 Theoriefragen

a Für welche Aufgaben in der Computerlinguistik eignen sich Listen, Dictionaries und Tupel in Python? Gib für jeden Typ 1-2 Beispiele an.

Datenstruktur	Beispiel
Listen	Listen sind nützlich, wenn die Reihenfolge der Elemente von Bedeutung ist und so oder so immer durch den kompletten Inhalt iterieren will. Beispiele dafür sind zum Beispiel die annotierten Wörter eines Satzes. Hier will man die Reihenfolge beibehalten, damit der Satz noch einen Sinn ergibt.
Dictionaries	Dictionaries sind hervorragend dafür geeignet, wenn man schnell auf einzelne Elemente zugreifen will und nicht an der Reihenfolge interessiert ist. Beispiele dafür sind das kleine Übersetzungsprogramm, das wir in der Stunde programmiert haben. Da wird schnell auf die einzelnen deutschen Wörter zugegriffen und die englische Version zurückgegeben. Ein weiteres Beispiel ist die Aufgabe 2 in Übung 4 in der zwei Texte verglichen werden sollen. Ohne Dictionaries dauert der Vergleich ein Vielfaches länger.
Tuple	Tuples sind nützlich wenn man kleinere Verbindungen von Elementen haben will. Ein Beispiel dafür ist die Tokenisierung. Man kann da jedem Wort eine Liste von Tags zuweisen.

b Welche Unterschiede gibt es in der Verwendung von regulären Ausdrücken in grep und in Python? Veranschauliche die Unterschiede an 2-3 Beispielen.

- Bei grep werden die gefundenen Strings direkt zurückgegeben, während man bei Python ein Objekt erhält, welches man zuerst noch entpacken muss.
- Bei grep können mehrere Ausdrücke mit '|' kombiniert werden, welche dann sequentiell ausgeführt werden. Dies ist in Python nicht möglich (zumindest wissen wir nicht wie)
- Bei grep können Filter angewendet werden. In Python gibt es keine.
- Bei grep muss der Input immer ein File sein. Bei Python immer ein String

Beispiele:

```
$ grep -E '^\w*[rR]\w*\t' * | wc
#| -> Kombination der Ausdrücke, wc -> Filter
```

```
re.search(r"[A-Z]+", str(tag)).group(0)
#re.search() ist vergleichbar mit grep -E 'Pattern'
#.group() wird dafür verwendet, das gefundene Objekt
#in einem String auszugeben
```

```
$ grep -E '^[A-Z]+' test.txt | wc
#test.txt beinhaltet "TEST"
#Output: 1 1 5
```

```
match = re.findall(r"[A-Z]+", str("TEST"))
count = len(match)
print count, count, len(match[0])
#Output: 1 1 4 (without newline)
```

c Welche Vor- und Nachteile haben verschiedene Codierungen, insbesondere UTF-8, Latin-1 und ASCII? Gebe je drei Vor- und Nachteile der drei Codierungen an, insbesondere im Bezug auf mehrsprachige Texte.

Codierung	Vorteil	Nachteil
UTF-8	Deckt grosses Gebiet an Sprachen ab (beinhaltet viele Sonderzeichen)	Braucht viel Speicherplatz
	Weit verbreitet (Benutzung)	Aufwändiger zu verarbeiten
	Kompatibel zu ASCII	Variable Länge der Zeichen. Viele kyrillische Zeichen werden aus mehreren Zeichen zusammengesetzt, was dazu führt das z.B. vélo 5 Zeichen lang ist statt 4.
Latin-1	Deckt die meisten nordeuropäischen Zeichen ab	Deckt nur kleines Gebiet an Sprachen ab
	Hat sich etabliert (schon lange verfügbar)	veraltet (zum Beispiel kein Eurozeichen)
	Braucht nicht allzu viel Speicherplatz	Kann auch nicht wirklich viel
ASCII	Braucht kaum Speicherplatz	Deckt nur 128 Zeichen ab
	Sehr simpel zu verarbeiten	Praktisch unbrauchbar für Sprachverarbeitung ausser Englisch
	Hat sich etabliert	veraltet

Reflexion/Feedback

a) Fasse deine Erkenntnisse und Lernfortschritte in zwei Sätzen zusammen.

Wir haben die Vorteile der verschiedenen Datenstrukturen gelernt. Ausserdem haben wir unsere Python-Kenntnisse noch weiter verbessern können.

b) Wie viel Zeit hast du in diese Übungen investiert?

ca. 5 Stunden