Python Programming and NLP (in English)

Pascual Martínez-Gómez
martinez.gomez.pascual@ocha.ac.jp

Ochanomizu University
National Institute of Informatics

December 8th

Summary

There are 4 main Data Structures in Python:

1. Lists.

Summary

There are 4 main Data Structures in Python:

1. Lists.
2. Tuples.

Summary

There are 4 main Data Structures in Python:

1. Lists.
2. Tuples.
3. Sets.

Summary

There are 4 main Data Structures in Python:

1. Lists.
2. Tuples.
3. Sets.
4. Dictionaries.

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
```

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
>>> t[2]
3
```

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
>>> t[2]
3
>>> t[2] = 30
Error!  (tuples are immutable)
```

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
>>> t[2]
3
>>> t[2] = 30
Error!  (tuples are immutable)
>>> r = (4,5,6)
```

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
>>> t[2]
3
>>> t[2] = 30
Error!  (tuples are immutable)
>>> r = (4,5,6)
>>> q = t + r
```

Summary

Tuples:

- Are very similar to lists.

```
>>> t = (1,2,3)
>>> t[2]
3
>>> t[2] = 30
Error!  (tuples are immutable)
>>> r = (4,5,6)
>>> q = t + r
>>> q (1,2,3,4,5,6)
```

Summary

Sets:

- Similar to lists, without duplicates.

```
>>> a = set([1,2,3,2])
>>> a
set([1,2,3])
```

Summary

Sets:

- Similar to lists, without duplicates.

```
>>> a = set([1,2,3,2])
>>> a
set([1,2,3])
>>> a[0]
Error!
```

Summary

Sets:

- Similar to lists, without duplicates.

```
>>> a = set([1,2,3,2])
>>> a
set([1,2,3])
>>> a[0]
Error!
>>> a.add(5)
>>> a
set([1,2,3,5])
```

Summary

Sets:

- Similar to lists, without duplicates.

```
>>> a = set([1,2,3,2])
>>> a
set([1,2,3])
>>> a[0]
Error!
>>> a.add(5)
>>> a
set([1,2,3,5])
>>> a.add(5)
>>> a
set([1,2,3,5])
```

Summary

Sets:

- Similar to lists, without duplicates.

Summary

Sets:

- Similar to lists, without duplicates.

```
>>> l = [1,2,3,2]
>>> b = set(l)
>>> b
[1,2,3]
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> d = {}
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> d = {}
>>> d = {'hello' : 'English', 'hola' :
'Spanish', 'hallo' : 'German'}
>>> d
{'hello' : 'English', 'hola' : 'Spanish',
'hallo' : 'German'}
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> d = {}
>>> d = {'hello' : 'English', 'hola' :
'Spanish', 'hallo' : 'German'}
>>> d
{'hello' : 'English', 'hola' : 'Spanish',
'hallo' : 'German'}
>>> d['hello']
'English'
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> d = {}
>>> d = {'hello' : 'English', 'hola' :
'Spanish', 'hallo' : 'German'}
>>> d
{'hello' : 'English', 'hola' : 'Spanish',
'hallo' : 'German'}
>>> d['hello']
'English'
>>> d['konnichiwa'] = 'Japanese'
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> d = {}
>>> d = {'hello' : 'English', 'hola' :
'Spanish', 'hallo' : 'German'}
>>> d
{'hello' : 'English', 'hola' : 'Spanish',
'hallo' : 'German'}
>>> d['hello']
'English'
>>> d['konnichiwa'] = 'Japanese'
>>> d
{'hello' : 'English', 'hola' : 'Spanish',
 'hallo' : 'German', 'konnichiwa' : 'Japanese'}
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> del d['hola']
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> del d['hola']
>>> d
{'hello' : 'English', 'hallo' : 'German',
'konnichiwa' : 'Japanese'}
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> del d['hola']
>>> d
{'hello' : 'English', 'hallo' : 'German',
'konnichiwa' : 'Japanese'}
>>> 'hola' in d
False
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> del d['hola']
>>> d
{'hello' : 'English', 'hallo' : 'German',
'konnichiwa' : 'Japanese'}
>>> 'hola' in d
False
>>> 'hello' in d
True
```

Summary

Dictionaries (my favourite):

- Store a *value* for each *key*.

```
>>> del d['hola']
>>> d
{'hello' : 'English', 'hallo' : 'German',
'konnichiwa' : 'Japanese'}
>>> 'hola' in d
False
>>> 'hello' in d
True
>>> d.keys()
['hello', 'hallo', 'konnichiwa']
```

Review

- Solution of exercise1 (seminar 7) is in Dropbox.

Review

- Solution of exercise1 (seminar 7) is in Dropbox.
- Do you have questions about the exercise?

Review

- Solution of exercise1 (seminar 7) is in Dropbox.
- Do you have questions about the exercise?
- Do you have questions about the review material?
  - Section 5.3 (tuples),
  - Section 5.4 (sets),
  - Section 5.5 (dictionaries).

Input-Output (IO) 1 / 1

- Open a file for reading or writing:

```
import codecs
```

Input-Output (IO) 1 / 1

- Open a file for reading or writing:

```
import codecs
finput = codecs.open('my_file.txt', 'r', 'utf-8')
finput.close()
```

Input-Output (IO) 1 / 1

- Open a file for reading or writing:

```
import codecs
finput = codecs.open('my_file.txt', 'r', 'utf-8')
for line in finput:
  print(line)
finput.close()
```

1. Loop over file.

Input-Output (IO) 1 / 1

- Open a file for reading or writing:

```
import codecs
finput = codecs.open('my_file.txt', 'r', 'utf-8')
lines = finput.readlines()
for line in lines:
  print(line)
finput.close()
```

2. Read all lines.

Input-Output (IO) 1 / 1

- Open a file for reading or writing:

```
import codecs
finput = codecs.open('my_file.txt', 'r', 'utf-8')
content_str = finput.read()
print(content_str)
finput.close()
```

3. Read whole string.

Exercises

- Exercise 1: Given a file with text, extract the vocabulary (unique words).

Exercises

- Exercise 1: Given a file with text, extract the vocabulary (unique words).
- Download file 'little_prince.txt' from DropBox.

Exercises

- Exercise 1: Given a file with text, extract the vocabulary (unique words).
- Download file 'little_prince.txt' from DropBox.
- Example:

```
filename = 'little_prince.txt'
```

Exercises

- Exercise 1: Given a file with text, extract the vocabulary (unique words).
- Download file 'little_prince.txt' from DropBox.
- Example:

```
filename = 'little_prince.txt'
unique_words = GetUniqueWords(filename)
print(unique_words)
...
```

Exercises

- Exercise 2: Given a file with text, find what is the most popular word and most popular character.

Exercises

- Exercise 2: Given a file with text, find what is the most popular word and most popular character.
- Example:
  `filename = 'little_prince.txt'`

Exercises

- Exercise 2: Given a file with text, find what is the most popular word and most popular character.
- Example:

```
filename = 'little_prince.txt'
popular_word = PopularWord(filename)
print(popular_word)
```

Exercises

- Exercise 2: Given a file with text, find what is the most popular word and most popular character.
- Example:

```
filename = 'little_prince.txt'
popular_word = PopularWord(filename)
print(popular_word)
...
popular_char = PopularChar(filename)
print(popular_char)
...
```

Exercises

- Exercise 3: Given a file with text, count how many characters, words and lines are there.

Exercises

- Exercise 3: Given a file with text, count how many characters, words and lines are there.
- Example:
  `filename = 'little_prince.txt'`

Exercises

- Exercise 3: Given a file with text, count how many characters, words and lines are there.
- Example:

```
filename = 'little_prince.txt'
counters = CountCharsWordsLines(filename)
print(counters)
(x, y, z)
```

Exercises

- Exercise 4: Given a file with text, count 2-grams (bi-grams).

Exercises

- Exercise 4: Given a file with text, count 2-grams (bi-grams).
- Example:
  `filename = 'little_prince.txt'`

Exercises

- Exercise 4: Given a file with text, count 2-grams (bi-grams).
- Example:

```
filename = 'little_prince.txt'
bigrams = CountBigrams(filename)
```

Exercises

- Exercise 4: Given a file with text, count 2-grams (bi-grams).
- Example:

```
filename = 'little_prince.txt'
bigrams = CountBigrams(filename)
print(bigrams)
(word1, word2, count)
...
```

Exercises

- Exercise 5: Given a file with text, count 2-grams (bi-grams) **probabilities**.

Exercises

- Exercise 5: Given a file with text, count 2-grams (bi-grams) **probabilities**.
- Example:

```
filename = 'little_prince.txt'
bigrams = CountBigrams(filename)
print(bigrams)
(word1, word2, probability)
...
```

Exercises

- Exercise 5: Given a file with text, count 2-grams (bi-grams) **probabilities**.
- Example:

```
filename = 'little_prince.txt'
bigrams = CountBigrams(filename)
print(bigrams)
(word1, word2, probability)
...
```

$$\Pr(\text{word2} \mid \text{word1}) = \frac{\text{count}(\text{word1}, \text{word2})}{\text{count}(\text{word1})} \qquad (1)$$

Homework

1. Please do Exercises 3, 4 and 5.

Homework

1. Please do Exercises 3, 4 and 5.
2. Please review:
   - Review section 7.2 (Reading and writing files), and
   - Review section 7.2.1 (Methods of file objects).
   - No need to review 7.2.2.

Homework

1. Please do Exercises 3, 4 and 5.
2. Please review:
   - Review section 7.2 (Reading and writing files), and
   - Review section 7.2.1 (Methods of file objects).
   - No need to review 7.2.2.

Contact:

- You can visit me on Mondays (any time) at 308.
  - Ask programming questions.
  - Ask about natural language processing.
  - Talk in English.
- You can also write me at:
  - `martinez.gomez.pascual@ocha.ac.jp`