Exercise Session 9:
Dictionaries and collections comprehensions

September 14th 2020

Dictionaries in python

A dictionary is a collection which is **unordered, changeable and indexed**.

In Python dictionaries are written with curly brackets, and they have keys and values separated by a ‘:’ character.

Keys are unique within a dictionary while **values may not be**. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

```
{‘eye_color’: ’green’, ‘fave_color’: ‘green’, 7: [2, True]}
```

```
{‘fave_color’: ‘green’, ‘fave_color’: ‘blue’, [2, True]: 7}
```

Exercise 9.1: Building a dictionary from scratch

Define an empty dictionary `myPet = {}`

Add the following items:

a. The name is ‘Spot’
b. The animal is ‘dog’
c. The age is 4
d. The favorite snacks are ‘sausages’, ‘peanut butter’, and ‘dropped popcorn’

When printed, the resulting dictionary should look like this:

```
{‘name’: ‘Spot’, ‘animal’: ‘dog’, ‘age’: 4, ‘fave_snacks’:
[‘sausages’, ‘peanut butter’, ’dropped popcorn’]}
```

Hints: *Python Dictionary Description (with examples)*

Exercise 9.2: Modifying an existing dictionary

```
inventory = {
    'gold' : 500,
    'pouch' : ['flint', 'twine', 'gemstone'],
    'backpack' : ['xylophone','dagger', 'bedroll','bread loaf']
    }
```

Paste the above dictionary to your text editor and perform the following operations:

a. Add a key to inventory called 'pocket'.
b. Set the value of 'pocket' to be a list consisting of the strings 'seashell' and 'lint'
c. Sort the items in the list stored under the 'backpack' key.
   Hint: list methods `.sort()`
d. Remove 'dagger' from the list of items stored under the 'backpack' key.
e. Add 50 to the number stored under the 'gold' key.

Hints: *Python Dictionary Methods*

Exercise 9.3: Dictionary applications, counts

Download the file JFK.txt from LearnIT.

a. Write a function that reads in the file and returns a list, of all the words in the text (one word per item). All words should be converted to lowercase.
   Hint: `.strip()`

b. Write a function that filters the list of all words and returns a new list containing a single instance of each unique word of the word list.
   Hint: use sets

c. Write a function that loops through the word list, counts the occurence of each word, and stores the results in a dictionary. Here, the keys of the dictionary will be all the unique words and the corresponding values will be their counts.

Collections comprehensions

Collections comprehensions are a **very concise** way to create collections. They are used to create a new collection by iterating over another collection.

Examples:

```
list1 = [i for i in range(4)]
>> [0,1,2,3]

list2 = [i for i in list1) if i+2 == 3]
>> [1]

dict1 = {i:[] for i in list1}
>> {0:[], 1:[], 2:[], 3:[]}

list3 = [x for x in dict1.keys()}
>> [0,1,2,3]
```

Exercise 9.4: Writing list comprehensions

a. Write a list comprehension that returns a list containing all the integers from 0 to 9.

```
>> [0,1,2,3,4,5,6,7,8,9]
```

Hint: `range()`

b. Write a list comprehension that returns the squares of all the numbers from your original list of integers.

```
>> [0,1,4,9,16,25,36,42,64,81]
```

c. Write a list comprehension that returns the cube of all the odd numbers from your list of integers.

```
>> [1,27,125,343,729]
```

Hints: *Python List Comprehension Syntax*

Exercise 9.5: Writing dictionary comprehensions

```
students = {"dennis": 23, "david": 21, "mary": 9, "daniel":
25, "darius": 17, "jim": 10, "marvin": 19}
```

For the given dictionary which contains the names and ages of people, perform the following:

a. Use dictionary comprehension to obtain a new dictionary, where all the names are converted to upper case.
   `>> {”DENNIS": 23, ”DAVID": 21, … , ”MARVIN": 19}`
   Hint: `.upper()`

b. Use dictionary comprehension to obtain a new list containing only the names of students who are eligible for voting. A person is eligible to vote if he/she is at least 18 years old.
   `>> ["dennis", "david", "daniel", "marvin”]`

Hints: *Python Dictionary Comprehension Syntax*

Further Resources

The following resources are completely optional suggestions in line with today's exercises for you to explore at home, in case you wish to strengthen your understanding of the material.

- *Official python documentation for dictionaries*
- *Datacamp: list comprehension tutorial*
- *Datacamp: dictionary comprehension tutorial*