Introduction to programming, Lesson 4: dictionaries, files, modules

Dictionaries

A dictionary is a data type similar to lists, but works with keys and values instead of indexes. Each value stored in a dictionary can be accessed using a key, which is any type of object (a string, a number, a list, etc.) instead of using its index to address it. For example, a database of phone numbers could be stored using a dictionary like this:

```
phonebook = {}
phonebook["John"] = 938477566
phonebook["Jack"] = 938377264
phonebook["Jill"] = 947662781
print(phonebook)
```

Alternatively, a dictionary can be initialized with the same values in the following notation:

```
phonebook = {
    "John" : 938477566,
    "Jack" : 938377264,
    "Jill" : 947662781
}
print(phonebook)
```

Dictionaries can be iterated over, just like a list. However, a dictionary, unlike a list, does not keep the order of the values stored in it. To iterate over key value pairs, use the following syntax:

```
phonebook = {"John" : 938477566,"Jack" : 938377264,"Jill" : 
947662781}
for name, number in phonebook.items():
    print("Phone number of %s is %d" % (name, number))
```

To remove a specified entry, use

```
del phonebook['John']
```

Exercise 1. Add "Jake" to the phonebook with the phone number 938273443, and remove Jill from the phonebook.

```
phonebook = {
    "John" : 938477566,
    "Jack" : 938377264,
    "Jill" : 947662781
}

# write your code here

# testing code
if "Jake" in phonebook:
    print("Jake is listed in the phonebook.")
if "Jill" not in phonebook:
    print("Jill is not listed in the phonebook.")
```

Introduction to programming, Lesson 4: dictionaries, files, modules

Reading from files, writing to files

```
file_name = "example.txt"
in_file = open(file_name,'r')#Open example.txt for reading
content = in_file.read()
print content
in_file.close()
```

Exercise 2. Create a file example.txt. In the same folder, create a script that reads example.txt and prints out the content of example.txt. Try to do the same thing when example.txt is in a different folder.

Sometimes it is more convenient to work with files line by line.

```
in_file = open(file_name,'r')#Open file_name for reading
lines = in_file.readlines()
for line in lines:
  # Do something with the line
in_file.close()
```

Exercise 3. Create a file example.txt, where each line contains an integer. In the same folder, create a script that reads example.txt and prints out all the integers in the file that belong to the interval [10,20].

Exercise 4. Fix this script so that it writes the name of each animal on a separate line. If the output file already exists and you want to add something to the end, you must use mode 'a' (append) instead of 'w'.

```
out_file = open(file_name,'w')#Open file_name for writing
animals = ['elephant', 'cat', 'dog']
for animal in animals:
  out_file.write(animal)
out_file.close()
```

Modules

Python developers have implemented many useful functions, and you should use them. Functions are grouped in collections called 'modules'. For example, there is a module math that contains different mathematical functions () , there is a module re that allows you to work with regular expressions (https://docs.python.org/2/library/re.html), and so on. You can also create your own modules – a module is a python file that contains a collection of functions. If the name of the python file is “economics.py”, the name of the module will be economics.

```
from math import *
root = sqrt(49)
angle = pi/6
print sin(pi/6)
```

Use module math to compute the largest integer smaller than 4.76. Write a module “integers” that will contain a single function floor. The function must receive a float number x and output the largest integer smaller than x.

Introduction to programming, Lesson 4: dictionaries, files, modules

Harder exercises

1. **“Me Too” flashmob**
Download the file https://bit.ly/2TItCWF. The file contains the data related to the “Me Too” movement, which spread virally as a hashtag used on social media to help demonstrate the widespread prevalence of sexual assault and harassment, especially in the workplace. The file was generated by Google Trends. Each line of the file is of form “week,popularity”. Popularity is a number indicating the popularity of queries related to the movement during a certain week of the year. Your task is to find the week when the movement was most popular. Use function split to break a line into week and popularity.

2. **A Hundred Thousand Billion Poems**

A Hundred Thousand Billion Poems (original French title: *Cent mille milliards de poèmes*) is a book by Raymond Queneau, published in 1961. The book is a set of ten sonnets printed on card with each line on a separate strip. As all ten sonnets have not just the same rhyme scheme but the same rhyme sounds, any lines from a sonnet can be combined with any from the nine others, allowing for 10^{14} (= 100,000,000,000,000) different poems. When Queneau ran into trouble creating the book, he solicited the help of a mathematician Francois Le Lionnais. Your task will be to write a script that generates a random poem from a similar book. Download this file: https://bit.ly/2F7X00v. It contains 14 groups of lines, and each group consists of 10 lines. Groups are separated by an empty line.

- Write a script that reads the file and generates a list of 14 lists, where each list corresponds to a group and contains 10 lines.
- Modify your script so that it returns a poem of 14 lines, where each line is chosen at random from each group. Use Python module `random`.

3. **Text generator**

- Write a function that generates a list of pairs of consecutive words in a text. Save this function in a module. Write a script that imports the module you have just created, receives a name of a file, reads the file, and prints out the pairs of consecutive words in the file.
- Add to your module a Python function that receives a text and generates a dictionary. Each key is a word of the text, and the value for this key is a list of all the words that follow the key in the text.
- Write a function that receives a name of a file, reads its content, and generates a new text at random. The first word W^1 of the new text is a random word of the old text. (Use function random() to choose it.) The second word W^2 is a random word in the list of words following W^1 in the old text, and so on. Your new text must be of the same length or shorter than the old text. Once you have finished generating the text, write it into a (new) file. You can test your file on The Gold-Bug novel by Poe, which is available on Moodle.
- (*) Change your generator so that triples of consecutive words in the new text are triples of consecutive words in the old text.

You can read more about automatic text generators on Wikipedia: https://en.wikipedia.org/wiki/Natural_language_generation

4. Linear interpolation

When we cover the numerical libraries, we will see they include many alternatives for interpolation and function approximation. Nevertheless, let's write our own function approximation routine as an exercise. In particular, without using any imports, write a function `linapprox` that takes as arguments

- A function `f` mapping some interval [*a*,*b*] into $\mathbb{R}$
- Two numbers `a` and `b` providing the limits of this interval
- An integer `n` determining the number of grid points
- A number `x` satisfying `a <= x <= b`

and returns the piecewise linear interpolation of `f` at `x`, based on `n` evenly spaced grid points `a = point[0] < point[1] < ... < point[n-1] = b`. Aim for clarity, not efficiency.