LINKÖPING UNIVERSITY
Department of Computer and Information Science
Division of Statistics
Johan Falkenjack

2014-03-29
Text Mining
732A47

Introduction to Python - Computer Lab

Deadline: October 29 at 6 pm
Teacher: Johan Falkenjack
Grades: Pass/Fail
Submission: By email to Johan Falkenjack

Write your solutions in COMPLETE, READABLE and EXECUTABLE code.
Solutions should be written in a text file with .py extension.
Graphs produced during the lab should NOT be submitted, submit only the code.
Comment directly in the code (using # or ”’) whenever something needs to be explained.

---

1. Strings
   (a) Define the variable `parrot` containing the sentence *It is dead, that is what is wrong with it.*
   (b) Count the number of characters (letters, blank space, commas, periods etc) in the sentence.
   (c) Write code that counts the number of *letters* in the sentence.
   (d) Separate the sentence into a list of words. Call the list `ParrotWords`.
   (e) Merge (concatenate) `ParrotWords` into a sentence again.

2. Loops and list comprehensions
   (a) Write a for loop that produces the following output on the screen:
   The next number in the loop is 5
   The next number in the loop is 6
   ...
   The next number in the loop is 10
   [Hint: the `range()` function has more than one argument].
   (b) Write a while-loop that repeatedly generates a random number from a uniform distribution over the interval $[0, 1]$, and prints the sentence 'The random number is smaller than 0.9' on the screen until the generated random number is smaller than 0.9. [Hint: Python has a `random` module with basic random number generators].
   (c) Write a for-loop that iterates over the list `names = ['Ludwig','Rosa','Mona','Amadeus']` and writes the following to the screen:

The name Ludwig is nice
The name Rosa is nice
...
The name Amadeus is nice

Use Python's string formatting capabilities (the `%s` stuff ...) to solve the problem.

(d) Write a for-loop that iterates over the list `names = ['Ludwig','Rosa','Mona','Amadeus']` and produces the list `nLetters = [6,4,4,7]` that counts the letters in each name. [Hint: the pretty version uses the `enumerate()` function]

(e) Solve the previous question using a list comprehension.

(f) Use a list comprehension to produce a list that indicates if the name has more than four letters. The answer should be `shortLong = ['long','short','short','long']`.

(g) Write a loop that *simultaneously* loops over the lists `names` and `shortLong` to write the following to the screen

The name Ludwig is a long name
The name Rosa is a short name
...
The next Amadeus is a long name

[Hint: use the `zip()` function and Python's string formatting.]

3. Dictionaries

(a) Make a dictionary named `Amadeus` containing the information that the student Amadeus is a male (M), scored 8 on the Algebra exam and 13 on the History exam.

(b) Make three more dictionaries, one for each of the students: Rosa, Mona and Ludwig, from the information in the following table:

	Sex	Algebra	History
Rosa	F	19	22
Mona	F	6	27
Ludwig	M	9	5

(c) Combine the four students in a dictionary named `students` such that a user of your dictionary can type `students['Amadeus']['History']` to retrive Amadeus score on the history test. [HINT: The values in a dictionary can be dictionaries]

(d) Add the new student Karl to the dictionary `students`. Karl scored 14 on the Algebra exam and 10 on the History exam.

(e) Use for-loop to print out the names and scores of all students on the screen. The output should look like something this (the order of the students doesn't matter):

Student Amadeus scored 8 on the Algebra exam and 13 on the History exam
Student Rosa scored 19 on the Algebra exam and 22 on the History exam
...

[Hints: Dictionaries are iterables. A really pretty solution involves the `.items()` method of a dictionary]

4. Vectors and arrays

   (a) Define two lists: `list1 = [1,3,4]` and `list2 = [5,6,9]`. Try `list1*list2`. Does it work?

   (b) Import everything from scipy (`from scipy import *`). Convert `list1` and `list2` into arrays (name them `array1` and `array2`). Now try `array1*array2`.

   (c) Let `matrix1` be a 2-by-3 array with `array1` and `array2` as its two rows. Let `matrix2` be a 3-by-3 array with elements 1, 2 and 3 on the diagonal. Try `matrix1*matrix2`. Why doesn't this work?

   (d) Compute the usual matrix product of `matrix1` and `matrix2`.

5. Functions

   (a) Write a function `CircleArea(radius)` that computes the area of a circle with radius `radius`. Call the function to show that it works. [Hint: the number π needs to be loaded from the `math` module]

   (b) Modify the `CircleArea` function so that it checks it the radius is positive and prints *The radius must be positive* to the screen if it is not. Also, if the radius is not positive the function should return `None`.

   (c) Now write another function `RectangleArea(base,height)` that computes the area of a rectangle. Put both functions in a text file named `Geometry.py`. Close the Python interpreter (or all of Spyder, if you prefer). Start the interpreter and load the two area functions from the module. Remember to set the PYTHONPATH so that Python can find your file. This is done in Spyder from the **Tools** menu and then choosing PYTHONPATH.

   (d) Now define another function in your `Geometry` module that computes the area of a triangle. Try to import the new function from the module. Why does it not work?
   [Hint: try `import imp` followed by `imp.reload(Geometry)`. **Update**: this does not seem to work on all systems, at least as easy as I intend it to be. Solution: skip this part, or switch to Linux :-)]

Have fun!