Exercise 5: Function Definitions and Frequency Distributions

Programming Techniques in Computational Linguistics 1

Deadline: 02.12.2021

Remarks on submission

- Always name your submissions as follows: olatusernames_(pcl1|pfl)_exercise0X.txt/pdf/py/zip.
  - Example: jbiden_dtrump_pcl1_exercise05.zip
- Learning partnerships in pairs are mandatory. **Both students have to upload the solutions on olat!**
- Every file you upload, including Python programs, must contain the authors' name and the date.
- Write many comments! They don't just help the tutors to understand what you're thinking, but can also help you to find and fix possible bugs.
- Please number the tasks on the submission sheet/Python programs the same as on the task sheet.

1 List comprehension to function

The following Python expression is given:

```
from nltk.book import text2
sorted([word.lower() for word in text2 if len(word)>4 and len(word)<12])
```

a) What value does this expression evaluate to? Describe it in natural language.

b) Explain why `text2=3, text2=[1,2,3]` and `text2=[[1,2,3,4,5,6,7], 'winter_wren']` generate an error.

c) Write a function that returns the same output.

d) What changes if you replace `text2` with `set(text2)`?

Please submit an executable Python program. Write the explanations as comments.

2 Function to list comprehension

The following Python code is given:

```
def exclude_consonants(text):
   non_consonants_list = []
   for char in text:
      if char not in ['b', 'c', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'n', 'p',
                      'q', 'r', 's', 't', 'v', 'w', 'x', 'y', 'z']:
         non_consonants_list.append(char.upper())
   return non_consonants_list
converted_text = exclude_consonants('studying computational linguistics rocks!!')
print(converted_text)
```

a) Describe the code in natural language.

b) How could the code be made more efficient? Do not use list comprehension in this point.

c) What is the output of `print(exclude_consonants('Studying computational linguistics rocks!!'))`? Is there any difference with respect to the proposed code?

d) Write a list comprehension that returns the same output.

e) In the above code, `exclude_consonants` is evaluated on a string (`'studying computational linguistics rocks!!'`). What is another object type that can be passed to the function? Propose a working example.

Please submit an executable Python program. Write the descriptions as comments.

3 List comprehensions

Write a function named `print_list_comprehension_results` which uses list comprehension in order to find tokens which fulfills the following conditions:

a) All words that end with *ity* and contain the letter *b* but do not contain the letter *m*.

b) All words that start with *u* (**case-insensitive**) and are shorter than five characters.

c) All words that contain the sequence of letters *man*, but do not start or end with it.

d) All words with start with *tan* (**case-sensitive**) and do not have the letter *g* at the fourth position.

e) All words containing at least three *a* and only one *e*.

Note that:

1. The function must have a parameter, which means that the function call must look like this: `print_list_comprehension_results(text1)`.
2. Use the function to investigate the first book in `nltk.book`[1]. NLTK is the only external module you can use.
3. All subtasks must be solved with list comprehension.

The output should contain the subtask letters, the lists of found items, and the lengths of the lists:

```
#a:
['flexibility', 'obscurity', 'benignity', 'ubiquity', 'invariability', 'durability',
'possibility', 'inability', 'ability', 'perceptibility', 'celebrity', 'inflexibility',
'probability', 'obliquity', 'liability', 'brevity'] 16
#b
...
```

Please submit an executable Python program.

[1] If you haven't done so at the first tutorial, please install NLTK by following the instructions on `http://www.nltk.org/install.html`. Please also install NLTK Data:

```
$ python3
>>> import nltk
>>> nltk.download("book")
```

4 Local and global variables

Comment the following code and explain what is going on.

```
def function1():
   sentence = 'I love you!'
   print(sentence)

sentence = 'I miss you!'
function1()
print(sentence)
```

Please submit an executable Python program.

5 Counting words

Anna and Kevin are learning Python. They should program a function that counts the number of whitespace separated tokens in a text and in the end prints this number out. Anna solved the problem quickly, but Kevin still struggles after 30 minutes. Anna wants to support Kevin and showed him the following randomly ordered deindented code lines:

```
count = 0
def word_count(text):
for word in line.split():
count += 1
for line in text:
print(count)
```

a) Order and indent the following lines to create a function that counts and prints (only at the end) the number of whitespace separated tokens.

b) Add a doc-string that describes what the function does.

c) If you call this function, and assign the result to a variable, what is the content of the latter?

Please submit an executable Python program.

6 Accuracy, precision, recall and F-measure

You got to know accuracy (A), precision (P), recall (R) and F-measure (F) either in ECL1 or in the tutorial. The corresponding formulas are as follows:

$$A = \frac{TP + TN}{TP + TN + FP + FN},$$

$$P = \frac{TP}{TP + FP},$$

$$R = \frac{TP}{TP + FN},$$

$$F = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} = \frac{2 \cdot P \cdot R}{P + R},$$

where TP are true positives, TN are true negatives, FP are false positives, and FN are false negatives. You can find a text file named *measurement_data.txt* in the assignment folder[2]. The file contains universal POS tag[3], TP, FP, TN, and FN values (in this order) on each line. A tabulator separates the values.

[2]The values arise from the tagger in `https://mybinder.org/v2/gh/simon-clematide/introduction-to-cl-uzh-2021/main?urlpath=lab%2Ftree%2Fecl1-notebooks%2Fex03.ipynb` (with some modifications).

[3]See `https://universaldependencies.org/u/pos/`

In this task, you should write a program that:

1. reads first the input file;
2. for each line, calculates and outputs A, P, R, and F (rounded to two decimal places after the decimal point) using well-organized functions. The functions should also cover the special case that one or multiple of the values is zero (i.e., set measure to zero);
3. returns the POSs having the highest A, P, R, and F.

The output should look like this:

```
POS      A    P    R    F
-----------------------------------
ADJ    0.83    0.88    0.92    0.9
⋮
Best accuracy: ADP 0.97
Best precision: PUNCT 1.0
Best recall: SYM 1.0
Best F-measure: ADP 0.98
```

Please submit an executable Python program.

Reflection/Feedback

a) Summarise your discoveries and your learning progress in two sentences.

b) How long have you worked on the tasks of this exercise?