Python Introduction for NLP Task

Natural Language Processing (NLP) is a field of computer science, which includes the relations of computer and human (natural) language. The goal of NLP study is to teach computer to understand the human language. Therefore, studying NLP oftenly requires us to do computer programming. We can use several programming languages such as: Java, C, Python, etc for programming NLP task. However, in this course I will encourage the students to use Python, since it has rich supports for text/string operations, analysis, and there are a lot of available NLP libraries such as NLTK.

In this crash course, I will provide some basics information on Python programming (hm, giving pointer on several useful links to be exact), and examples of using NLTK library. I divide the topic into several parts:

1. Installing Python
2. Basics on Python programming
3. String operations
4. Data structure
5. NLTK basics
6. Exercise

1. Installing Python

Please visit https://wiki.python.org/moin/BeginnersGuide/Download to get information on which version you should install, etc. You can find 2 Python version, version 2 and version 3. You may also find several different functions on both version, but generally both of them provide similar functions. Usually, we need to check the libraries that we need in order to decide which version is more suitable to be installed. Since NLTK could be installed and run in Python 2 and Python 3, the preference is up to you. You can check https://wiki.python.org/moin/Python2orPython3 as a guide to choose the version.

Another option? Please check Anaconda https://anaconda.org/anaconda/python !

2. Basic of Python programming

Since I think most of you are familiar with Pascal, C, and Java, first I will highlight the syntax writing rule differences that we find on Python:

- There is no bracket to group the command.
- There is no semicolon to end a statement.
- You will use whitespace for indentation (2, 4, or 8 whitespaces)
- Python is untyped

As an interpreter, after entering the Python prompt, you can execute the commands by typing it on the Python command prompt. For example:

```
(D:\anaconda) D:\NLP_class>python
Python 3.6.0 |Anaconda 4.3.0 (64-bit)| (default, Dec 23 2016, 11:57:41) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World!")
Hello World!
```

However, when we need more complicated or longer line of codes, of course it would be more practical if we use a text editor (e.g: Notepad++) of even an IDE (e.g: PyCharm) to create the program file, and then run it from Python prompt.

We write Python programs on a block structure. A block is a piece of Python program text that is executed as a unit. The following are blocks: a module, a function body, and a class definition. A block is identified by its indentation.

As for the whole structure, if you are familiar with Java, you will remember the import statements-class block-main block structure. You can do the same for Python program by writing the import package on the beginning of the code, followed by the sequence of program blocks, and main definition. However, in order to execute the program, you do not need to always write the main definition (Please see http://gawron.sdsu.edu/python_for_ss/course_core/book_draft/anatomy/name_space.html and https://stackoverflow.com/questions/4041238/why-use-def-main for explanation) .

3. String Operations

String literals can be enclosed by either double or single quotes, although single quotes are more commonly used. Python provides rich methods on string manipulation, and we can access a string as a list of characters. For further explanation and examples, please visit https://developers.google.com/edu/python/strings .

4. Data structure

On writing more complex program, we might need the compound data type, to group several values. List (or you can say array, because you can access the element by its indices) is the most versatile compound data type in Python. Please visit https://docs.python.org/3/tutorial/introduction.html#lists for further explanations and several examples of list operation.

Other than list, there are other compound data types such as: set and dictionary. Please visit https://docs.python.org/3/tutorial/datastructures.html to read the explanation, including how to access the elements using loop.

5. NLTK basics

NLTK is a leading Python library used in natural language field. It has rich supports for language processing. For example, you can tokenize (split) the text, get the TAG information (e.g the POSTag), display the sentence parse tree, etc. You can access a bunch of dataset or corpora also, including Bahasa Indonesia dataset.

To use NLTK, of course you need to install it. The prerequisite of NLTK installation is Python, so make sure you already finish the Python installation. Please visit http://www.nltk.org/install.html as a guide on how to install NLTK, and http://www.nltk.org/data.html on how to add the data.

6. Example

- sentence tokenization

First, you need to specify the libraries you need. In this example, we need NLTK.

```
import nltk
```

Then we will try to tokenize a sentence. Tokenize means split a sentence into its elements (word/token).

#define the sentence

```
sentence = 'I am going to take the NLP class next semester'
```

#call the tokenize function

```
tokens = nltk.word_tokenize(sentence)
```

#iterate the tokens, and print its length

[left as an exercise]

7. Exercise

- Token Lemmatization. What is the difference between token lemmatization and token stemming? Could you perform stemming by using NLTK?

- Sentence tokenization, but try to use a sentence in Bahasa Indonesia. Do you notice any difference? Do you think the tokenization function could run successfully on any language?

- You can modify the exercise by playing with file Input/Output operation. For example, the input sentence is given as a text file, and the result will be output to a text file also.