GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Lecture 10: Repetition

Introduction for Linguists (LT2102)

Markus Forsberg
Språkbanken
University of Gothenburg

October 19, 2010

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Assignment 3

- **Revision**: `word_tokenize` and `sent_tokenize` are too broken for non-English texts. To tokenize into words, use, for all languages:

```
words = [token for token in nltk.wordpunct_tokenize(text)
                if token.isalpha()]
```

- **Clarification**: you should consider all N-grams of the language profiles when calculating the out-of-place (but only the top 300 N-grams of the profile of the unknown text).

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

About the assignments

- Because some of you arrived late, and others are really struggling, then I decided to accept assignment submissions until December 15 (last date to get a pass).
- If you are unable to complete them before that, then you have to return next time the course is given.
- However, submissions outside of the course must be complete solutions, and I will not be able to give you much help or quick responses.
- And do not make the mistake of prioritize reading for the exam instead of working on the assignments! Doing the assignments is the best way to prepare yourself for the exam.

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Programming and the coming courses

- You are supposed to learn the basics of programming in this course to be prepared for the coming courses.
- However, you will continue to practice programming in the coming courses, so you are not assumed to be programming wizards.

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question: exam preparation

- You need to practice to write programs without the constant feedback of the Python interpreter.
- You need to practice on convincing yourself that your program is correct (this is always important, but more so when you are not able to test it in the interpreter).
- Note: if you find it difficult to convince yourself that a function is correct, then you are probably trying to do to much in that function.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Question

- `x = raw_input ("Enter name: ")` =>
  I'm not sure where I should use "raw_input".

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question

- Exercise: define a function del/remove.

```
def remove(x,lst):
    for index in range(len(lst)):
        if lst[index] == x:
            return lst[:index] + lst[index+1:]
    raise ValueError, 'remove(x,lst): x not in list'
```

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question

- String formatting ⇒ when shall we use it? some of the cases are very basic i.e. we have to replace two "words" with another word (is this what we're supposed to know about string formatting? does it suffice? can you provide some examples)

```
>>> n = 1
>>> word = 'Honorificabilitudinitatibus'
>>> length = 27

>>> '%i. %s (%i)' % (n, word, length)
'1. Honorificabilitudinitatibus (27)'

compared with:

>>> str(n) + '. ' + word + ' (' + str(length) + ')'
'1. Honorificabilitudinitatibus (27)'
```

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question

- Maybe one or couple of examples with "while loop".

```
candy = []
while account > 0:
    account -= 100
    candy.append(buy_candy(100))

def divide_by_two_count(n):
    count = 0
    while n > 0:
        n = n/2
        count += 1
    return count
```

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question

- Consider the definition of "overlapping":

```
def overlapping(a1, a2):
    for x in a1:
        for y in a2:
            if x== y:
                return True
    return False
```

- Question is, when shall we write "return True" and the next one "return False"? why do we use 2 returns? and why not return True and the next one else: return False?

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Question: Classes

- Object-orientation is in the advanced part of the course.
- Classes: user-defined types.
- Inheritance: use another class (parent) as a template of the current class (child).
- The child class inherents the parents class methods and attributes, which may be overridden by the child class.
- An object of a class is an instance of that class.
- All classes inherents from an abstract class `object` (since this is the case, it is not required that we make it explicit).

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Class example

```
class Person(object):

    def __init__(self,firstname,lastname):
        self.firstname = firstname
        self.lastname = lastname

    def can_fly(self):
        return False

    def __str__(self):
        return '%s %s' % (self.firstname,self.lastname)

>>> p = Person(firstname='markus',lastname='forsberg')
>>> p.firstname
'markus'
>>> p.lastname
'forsberg'
>>> p.can_fly()
False
>>> print p # calls __str__, same as print str(p)
markus forsberg
```

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Class example (cont.)

```
class BirdPerson(Person):

    def can_fly(self):
        return True
```

```
>>> bp = BirdPerson(firstname='ducky',lastname='birdson')
>>> bp.firstname
'ducky'
>>> bp.lastname
'birdson'
>>> bp.can_fly()
True
>>> print bp
ducky birdson
```

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

1

Define a function `min()` that takes two numbers as arguments and returns the smallest of them. Use the if-then-else construct available in Python. (It is true that Python has the `min()` function built in, but writing it yourself is nevertheless a good exercise.)

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Define a function *min_of_three*() that takes three numbers as arguments and returns the smallest of them.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

3

Define a function that computes the length of a given list or string. (It is true that Python has the `len()` function built in, but writing it yourself is nevertheless a good exercise.)

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a function that takes a character (i.e. a string of length 1) and returns True if it is a vowel, False otherwise.

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

5

Write a function `translate()` that will translate a text into "rövarspråket" (Swedish for "robber's language"). That is, double every consonant and place an occurrence of `"o"` in between. For example, `translate("this is fun")` should return the string `"tothohisos isos fofunon"`.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Define a function `sum()` and a function `multiply()` that sums and multiplies (respectively) all the numbers in a list of numbers. For example, `sum([1, 2, 3, 4])` should return `10`, and `multiply([1, 2, 3, 4])` should return `24`.

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Define a function `reverse()` that computes the reversal of a string. For example, `reverse("I am testing")` should return the string `"gnitset ma I"`.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

8

Define a function *is_palindrome*() that recognizes palindromes (i.e. words that look the same written backwards). For example, *is_palindrome*("*radar*") should return True.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a function *is_member*() that takes a value (i.e. a number, string, etc) x and a list of values a, and returns True if x is a member of a, False otherwise. (Note that this is exactly what the in operator does, but for the sake of the exercise you should pretend Python did not have this operator.)

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

10

Define a function *pad_with_n_chars*() that takes a string `s`, an integer `n`, and a character `c` and returns a string consisting of `s` padded with `c` to create a string with a centered `s` of length `n`. For example, *pad_with_n_chars*("*dog*", 5, "*x*") should return the string `"xdogx"`.

GÖTEBORGS UNIVERSITET

Språk-BANKEN

CLT

Define a procedure (i.e. a function that doesn't return anything) `histogram()` that takes a list of tuples of strings and integers and prints a histogram to the screen. For example, `histogram([("dog",4), ("cat",9), ("mellow",7)])` should print the following:

```
dog: ****
cat: *********
mellow: *******
```

Refine the function to get the following output:

```
dog:     ****
cat:     *********
mellow: *******
```

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

The function `min()` from exercise 1) and the function *min_of_three*() from exercise 2) will only work for two and three numbers, respectively. But suppose we have a much larger number of numbers, or suppose we cannot tell in advance how many they are? Write a function *min_in_list*(), which takes a list of numbers and returns the smallest one. Use the function `min` your previously defined.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a function *find_shortest_word*() that takes a list of words and returns the length of the shortest one.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a function *filter_long_words*() that takes a list of words and an integer n and returns the list of words that are longer than n.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Represent a small bilingual lexicon:

```
merry : god
christmas : jul
and : och
happy : gott
new : nytt
year : år
```

and use it to translate your Christmas cards from and to English and Swedish. That is, write a function `translate()` that takes a list of words, and a string `"English"` or `"Swedish"` that defines the language of the list of words, and returns a list of words of the other language.

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a function *char_freq*() that takes a string and builds a frequency listing of the characters contained in it. Represent the frequency listing as a Python dictionary. Try it with something like *char_freq*(" *abbabcbdbabdbdbababababcbcbab*").

GÖTEBORGS
UNIVERSITET

Språk-
BANKEN

CLT

Write a procedure *char_freq_table*() with no parameters that, when run in a terminal, accepts a file name from the user, builds a frequency listing of the characters contained in the file, and prints a nicely formatted character frequency table to the screen.