Regular Expressions in Python

L435/L555

Dept. of Linguistics, Indiana University

Fall 2016

Regular expression module

Module

In order to use regular expressions, we need to load the module.

```
import re
```

Regular expression symbols

.	wildcard
\	escapes specials characters
[. . .]	character set
[∧ . . .]	complement of character set
|	or
*	Kleene star: 0 or more (of previous)
+	Kleene plus: 1 or more (of previous)
{ m,n }	repeat between m and n times
∧	beginning of a string
$	end of a string

Understanding regular expressions

See slides 28–37 here: http://cl.indiana.edu/~md7/16/245/slides/04-searching/slides.pdf

Regular expression functions

compile(<pattern>)	compiles a regex pattern into a pattern object – for reuse
search(<pattern>, <string>)	searches for regex pattern in string
match(<pattern>, <string>)	checks at **beginning** of string
split(<pattern>, <string>)	splits the string based on pattern, returns a **list**
findall(<pattern>, <string>)	returns a list of all occurrences
sub(<pat>, <rep>, <string>)	replaces pat by rep in string

Example

```
import re

mysent = input('Give me a sentence!\n')
if (not re.search('[ !\.;?]', mysent)):
    print('this is not a sentence')
```

Example

```
import re

mysent = input('Give me a sentence!\n')
newstr = re.sub('[A-Z]', 'XX', mysent)
print(newstr)
```

Pattern objects

Module

The functions `compile`, `search`, and `match` return a pattern object. The objects contain information about the pattern itself and for the matching functions also information about the matched segments in the string.

```
import re

phoneNums = re.compile('^\(?\d{3}[-␣)]\d{3}[␣-]\d{4}$')
myphone = input('Give␣me␣a␣phone␣number:␣')
if phoneNums.search(myphone):
    print('format␣correct')
else:
    print('format␣incorrect')
```

Example

```
import re

mysent = ' a rose is a rose is a rose'
allstr = re.search('( . )', mysent)
print(allstr.group(1))

allstr = re.findall('( . )', mysent)
print(allstr)
```