

Ling 555 — Programming for Linguists

Python — Strings

Robert Albert Felty

Speech Research Laboratory
Indiana University

Sep. 24, 2008

Templates

String basics

templates

width and precision

alignment

tuples

String methods

Definition

The string module provides templating functionality. Templates can be useful for internationalization and markup formats.

Example

The first heading in your “hello, world” webpage.

```
from string import Template
s=Template('<h1>$hello</h1>')
German=True
if German==True:
    print(s.substitute(hello='Servus, welt'))
else:
    print(s.substitute(hello='Hello, world'))
```

Width and precision

String basics

templates

width and precision

alignment

tuples

String methods

Definition

You can define how you want numbers to be printed out. Python's string formatting uses the same conventions as C.

practice

- 1 Print out pi to the 3rd decimal place, with a width of 7

```
print('%7.3f' % pi )
```
- 2 Print pi times 100 in the same fashion

```
print('%7.3f' % (100 * pi) )
```

Alignment

String basics

templates

width and precision

alignment

tuples

String methods

Practice

- 1 Print out 23 padded with zeros to make it 4 wide
`print('%04.0f' % 23)`
- 2 Print out 456.7 left aligned
`print('%-.1f' % 456.7)`

String and tuples

String basics

templates

width and precision

alignment

tuples

String methods

Example

You can format tuples all at once, e.g.

```
'%s: %4.2f' % ('pi', pi)
```

practice

- 1 Print out e and π , with appropriate labels, as in the preceding example, using a tuple

```
'%s: %4.2f, %s: %4.2f' % ('pi', pi,  
'e', e)
```

find

String basics

String methods

find

join / split

lower

replace

strip

translate

practice

- 1 Find where *in* starts in the phrase *needle in a haystack*

```
phrase='needle in a haystack'  
phrase.find('in')
```
- 2 If *needle in a haystack* contains *hay*, print *hey*

```
if phrase.find('hay')>=0:  
    print('hey')
```

Join and split

String basics

String methods

find

join / split

lower

replace

strip

translate

practice

- 1 Split the haystack phrase into multiple words
`words=phrase.split()`
- 2 Reverse the order of the words
`words.reverse()`
- 3 Join the words back together with commas
`','.join(words)`

Changing case

String basics

String methods

find

join / split

lower

replace

strip

translate

practice

- 1 Make *ALLCAPS* all lowercase
`'ALLCAPS'.lower()`
- 2 Make the first letter of *ALLCAPS* lowercase
`'ALLCAPS'.title()`

Replace

String basics

String methods

find

join / split

lower

replace

strip

translate

practice

- 1 Replace *needle* with *noodle* in the haystack phrase
`phrase.replace('needle', 'noodle')`
What is the value of `phrase` now?
- 2 Replace *e* with *o* in the haystack phrase
`phrase=phrase.replace('e', 'o')`

Strip

String basics

String methods

find

join / split

lower

replace

strip

translate

practice

- 1 Strip off newline characters from end of the haystack phrase
`phrase=phrase.rstrip('\r\n')`
- 2 Strip off any leading or trailing whitespace from the haystack phrase, and convert to upper case
`phrase=phrase.strip().upper()`
- 3 Strip off any leading or trailing whitespace from the haystack phrase, replace *needle* with *noodle* and convert to upper case
`phrase=phrase.strip().replace('needle', 'noodle').upper()`

Translate

String basics

String methods

find

join / split

lower

replace

strip

translate

Definition

Translate can be used to map an entire character set to a different one, using a one-to-one mapping.

Example

I accidentally switch my keyboard to German mode, in which *y* and *z* are switched, and type my entire thesis this way without noticing.

```
thesis='verbositz in verbaliying is  
      uglz'  
from string import maketrans  
germanToEnglish=maketrans('yz','zy')  
thesis.translate(germanToEnglish)
```