# Python String Manipulation Examples: Search, Replace, Wrap Strings

Fan Wang

2020-06-17

## Contents

## 1 Strings

Go to the **RMD**, **PDF**, or **HTML** version of this file. Go back to fan's Python Code Examples Repository (bookdown site).

```
import numpy as np
import string as string
import random as random
```

### 1.1 Generate Random Strings

Generate some random strings:

```
random.seed(123)
it_word_length = 5
st_rand_word = ''.join(random.choice(string.ascii_lowercase) for i in range(it_word_length))
st_rand_word = st_rand_word.capitalize()
print(f'{st_rand_word=}')
```

```
## st_rand_word='Bicyn'
```

Generate a block or random text and then convert it to a one list of strings:

```
random.seed(123)
it_words_count = 15
it_word_length = 5
st_rand_word_block = ''.join(random.choice(string.ascii_lowercase) for ctr in range(it_word_length*it_w
ls_st_rand_word = [st_rand_word_block[ctr: ctr + it_word_length].capitalize()
                   for ctr in range(0, len(st_rand_word_block), it_word_length)]
print(f'{ls_st_rand_word=}')
```

```
## ls_st_rand_word=['Bicyn', 'Idbmr', 'Rkkbf', 'Ekrkw', 'Hfany', 'Ctmca', 'Kxodb', 'Cveez', 'Ajnsp', 'I
```

Reshape the array of words to a matrix:

```
mt_st_rand_word = np.reshape(ls_st_rand_word, [3,5])
print(f'{mt_st_rand_word=}')
```

```
## mt_st_rand_word=array([['Bicyn', 'Idbmr', 'Rkkbf', 'Ekrkw', 'Hfany'],
##        ['Ctmca', 'Kxodb', 'Cveez', 'Ajnsp', 'Ipbyj'],
##        ['Kqzpg', 'Tuqsz', 'Kamyu', 'Qnvru', 'Zvtpq']], dtype='<U5')
```

```
print(f'{mt_st_rand_word.shape=}')
```

```
## mt_st_rand_word.shape=(3, 5)
```

```
print(f'{type(mt_st_rand_word)=}')
```

```
## type(mt_st_rand_word)=<class 'numpy.ndarray'>
```

### 1.2 Print Strings with Numeric Values and Other Strings

After some code segment, print some outputs declaring the end of operation and print results.

```
dc_invoke_main_args = {'speckey': 'ng_s_t',
                       'numeric': 1.46,
                       'ge': False,
                       'multiprocess': False}

print(f'speckey in dc_invoke_main_args is {dc_invoke_main_args["speckey"]}.')
```

```
## speckey in dc_invoke_main_args is ng_s_t.
```

```
print(f'numeric in dc_invoke_main_args is {dc_invoke_main_args["numeric"]}.')
```

```
## numeric in dc_invoke_main_args is 1.46.
```

```
print(f'speckey in dc_invoke_main_args is {dc_invoke_main_args}.')
```

```
## speckey in dc_invoke_main_args is {'speckey': 'ng_s_t', 'numeric': 1.46, 'ge': False, 'multiprocess'
```

### 1.3 Add String Suffix to Numeric Array

Given an numeric array, add string, for example to generate sequencial column names with suffix c:

```
ar_st_colnames = [ 's' + str(it_col) for it_col in np.array(range(1, 3))]
print(ar_st_colnames)
```

```
## ['s1', 's2']
```

### 1.4 Search if Names Include Strings

Given a list of strings, loop but skip if string contains elements string list.

```
# define string
ls_st_ignore = ['abc', 'efg', 'xyz']
ls_st_loop = ['ab cefg sdf', '12345', 'xyz', 'abc xyz', 'good morning']

# zip and loop and replace
for st_loop in ls_st_loop:
  if sum([st_ignore in st_loop for st_ignore in ls_st_ignore]):
    print('skip:', st_loop)
  else:
    print('not skip:', st_loop)
```

```
## skip: ab cefg sdf
## not skip: 12345
## skip: xyz
## skip: abc xyz
## not skip: good morning
```

## 1.5 Replace a Set of Strings in String

Replace terms in string

```
# define string
st_full = """
abc is a great efg, probably xyz. Yes, xyz is great, like efg.
eft good, EFG capitalized, efg good again.
A B C or abc or ABC. Interesting xyz.
"""

# define new and old
ls_st_old = ['abc', 'efg', 'xyz']
ls_st_new = ['123', '456', '789']

# zip and loop and replace
for old, new in zip(ls_st_old, ls_st_new):
  st_full = st_full.replace(old, new)

# print
print(st_full)
```

```
##
## 123 is a great 456, probably 789. Yes, 789 is great, like 456.
## eft good, EFG capitalized, 456 good again.
## A B C or 123 or ABC. Interesting 789.
```

## 1.6 Wrap String with Fixed Width

Given a long string, wrap it into multiple lines with fixed width.

```
import textwrap

# A long Path
st_path = """
C:/Users/fan/Documents/Dropbox (UH-ECON)/Project Emily Minority Survey/EthLang/reg_lang_abi_cls_mino/tab
"""

# Wrap text with tight width
st_wrapped = textwrap.fill(st_path, width = 20)
print(st_wrapped)
```

```
##  C:/Users/fan/Docume
## nts/Dropbox (UH-
## ECON)/Project Emily
## Minority Survey/EthL
## ang/reg_lang_abi_cls
## _mino/tab3_fm/attain
## _m_vs_f/tab3_mand_ta
## lk_m2c_hfracle02.tex
```

Combine Strings that are wrapped and not Wrapped

```
# Paths
st_path_a = "C:/Users/fan/Documents/Dropbox (UH-ECON)/Project Emily Minority Survey/EthLang/reg_lang_ab
st_path_b = 'C:/Users/fan/R4Econ/support/development/fs_packaging.html'

# Combine Strings and Wrap
str_dc_records = 'First Path:'.upper() + '\n' + \
                 textwrap.fill(st_path_a, width=25) + '\n\n' + \
                 'Second Path:'.upper() + '\n' + \
                 textwrap.fill(st_path_b, width=25)

# Print
print(str_dc_records)
```

```
## FIRST PATH:
## C:/Users/fan/Documents/Dr
## opbox (UH-ECON)/Project
## Emily Minority Survey/Eth
## Lang/reg_lang_abi_cls_min
## o/tab3_fm/attain_m_vs_f/t
## ab3_mand_talk_m2c_hfracle
## 02.tex
##
## SECOND PATH:
## C:/Users/fan/R4Econ/suppo
## rt/development/fs_packagi
## ng.html
```