Workshop 4

COMP20008

Elements of Data Processing
Zijie Xu

Agenda

- Data scraping
- Regular expressions

Data scraping

- Extracting data from websites or other sources on the internet
- Python package: `BeautifulSoup`
- Core idea:
  - Identify source document structure
  - Locate the things to extract
  - Perform relevant clean-ups to normalise data

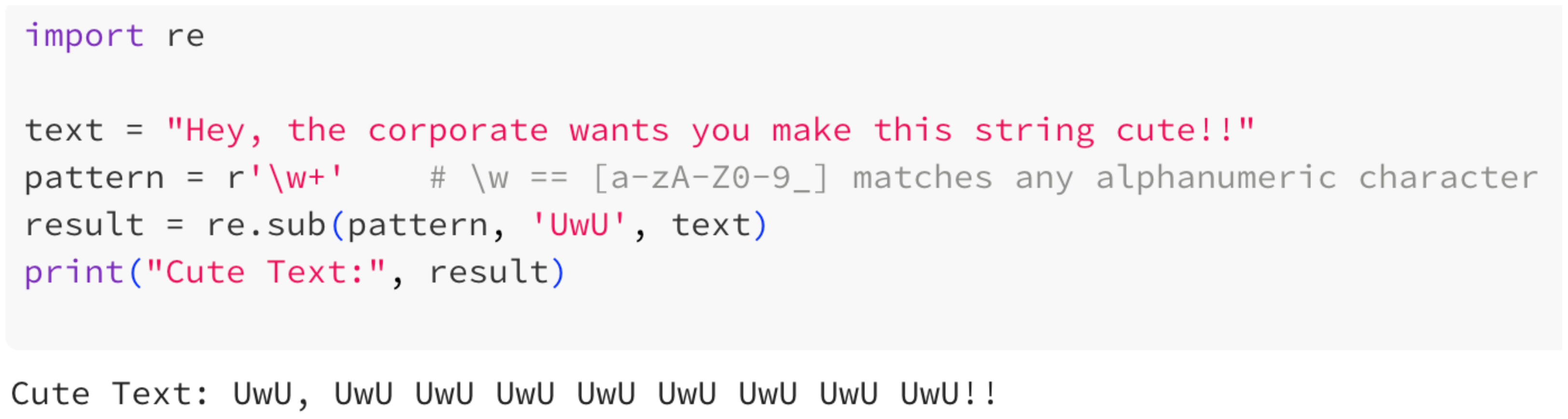
Regular Expressions

- Regular Expressions (RegEx) enable searching, matching, and manipulation of strings based on defined **search patterns**
  - Python `re` module: API and Tutorial
- Some useful methods
  - `re.search(`*`pattern`*`, `*`string`*`)`
  - `re.findall(`*`pattern`*`, `*`string`*`)`
  - `re.sub(`*`pattern`*`, `*`replacement`*`, `*`string`*`)`
  - `re.split(`*`pattern`*`, `*`string`*`)`

Regular expression

Example: `re.sub(pattern, replacement, string)`

A function from the `re` module that searches for occurrences of `pattern` in `string` and replaces them with `replacement`



```python
import re

text = "Hey, the corporate wants you make this string cute!!"
pattern = r'\w+'    # \w == [a-zA-Z0-9_] matches any alphanumeric character
result = re.sub(pattern, 'UwU', text)
print("Cute Text:", result)
```

```
Cute Text: UwU, UwU UwU UwU UwU UwU UwU UwU UwU!!
```

Regular Expressions

- Metacharacters `. ^ $ * + ? { } [ ] \ | ( )`
  - Wildcard

`.`	Matches any character

  - Anchor

`^`	Start of string
`$`	End of string

  - Repeats

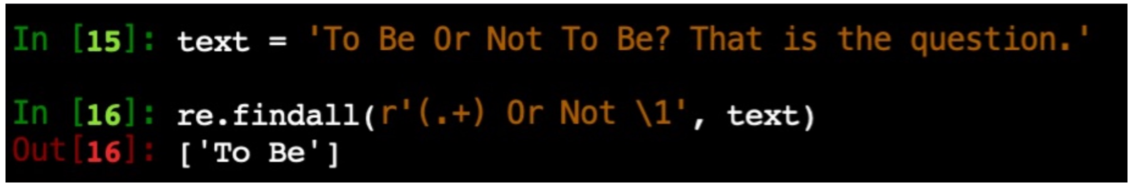
`*`	≥ 0
`+`	≥ 1
`?`	0 or 1
`{m,n}`	$m \leq$ # repeat $\leq n$

Regular Expressions

- Metacharacters
  - Character class/set
    - **`[ ]`** matches any character from a class of characters
    - **`[^]`** '^' as **first character** for complementing class
    - Metacharacters (except \) do not work in classes and will be matched as literals
    - Some predefined classes: **`\w \W \d \D \s \S`**

Regular Expressions

- Alternation
  - `|` split alternative patterns
- Capture groups
  - `( )` captures the matched part for later reference



```
In [15]: text = 'To Be Or Not To Be? That is the question.'

In [16]: re.findall(r'(.+) Or Not \1', text)
Out[16]: ['To Be']
```

- Lookahead assertions
  - `x(?=y)` matches x only if it is followed by y
  - `x(?!y)` matches x only if it is not followed by y
  - `y` is not in matched pattern

Regular Expressions

- \
  - Escapes metacharacters

    Use raw strings to avoid typing many double backslashes

    `'\\$' == r'\$'`
  - Escapes the name of a character class `\d \w`
  - Back-references a sequence captured by a capture groups `\1 \2`

THE UNIVERSITY OF MELBOURNE

Thank you

More Resources: Canvas