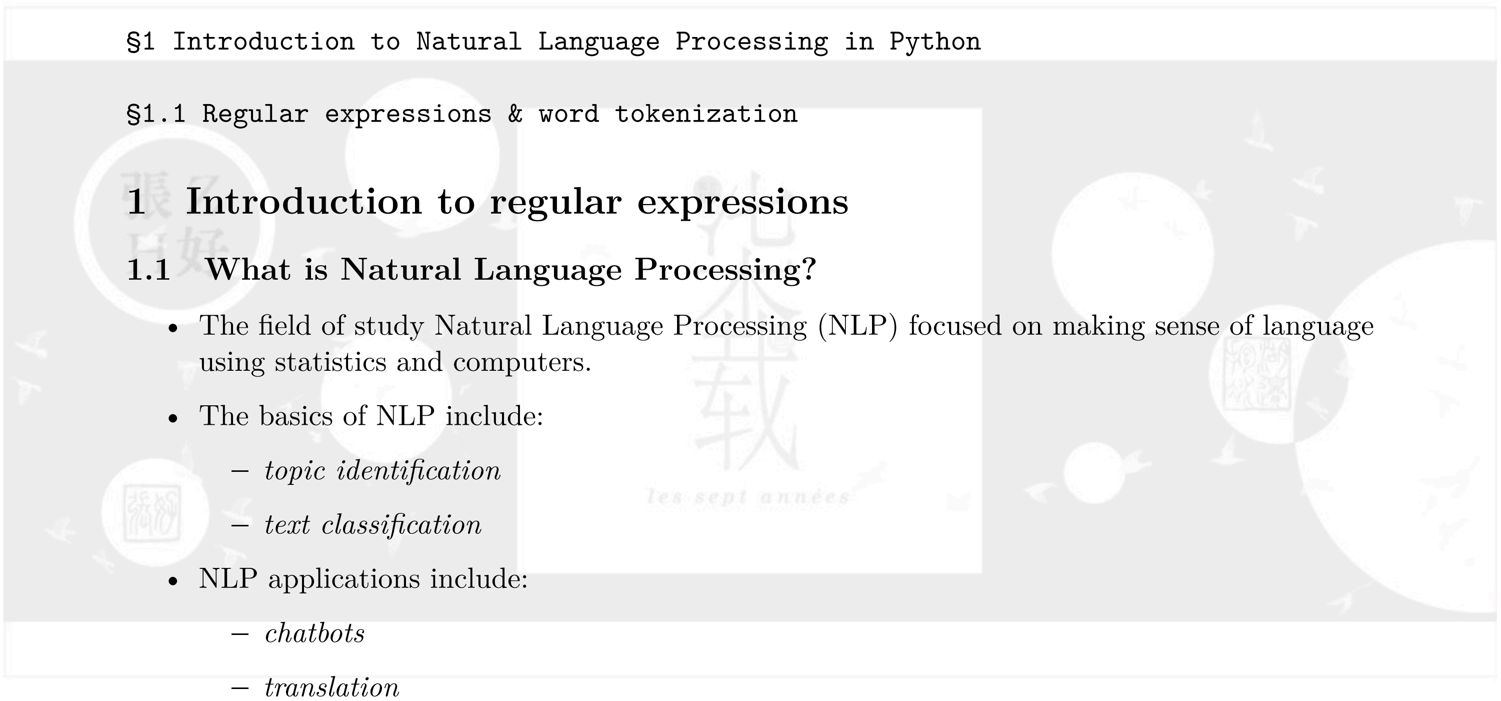
Introduction to regular expressions

LaTeX, Puteaux, 2020, 2021

```
###############################################
##                                           ##
##  Natural Language Processing in Python  ##
##                                           ##
###############################################
```



```
§1 Introduction to Natural Language Processing in Python

§1.1 Regular expressions & word tokenization
```

1 Introduction to regular expressions

1.1 What is Natural Language Processing?

- The field of study Natural Language Processing (NLP) focused on making sense of language using statistics and computers.
- The basics of NLP include:
  - *topic identification*
  - *text classification*
- NLP applications include:
  - *chatbots*
  - *translation*
  - *sentiment analysis*
  - *and many more*

1.2 What exactly are regular expressions?

- Strings with a special syntax
- Allow matching patterns in other strings, e.g.,
  - *find all web links in a document*
  - *parse email addresses*
  - *remove/replace unwanted characters*

1.3 Code of the applications of regular expressions:

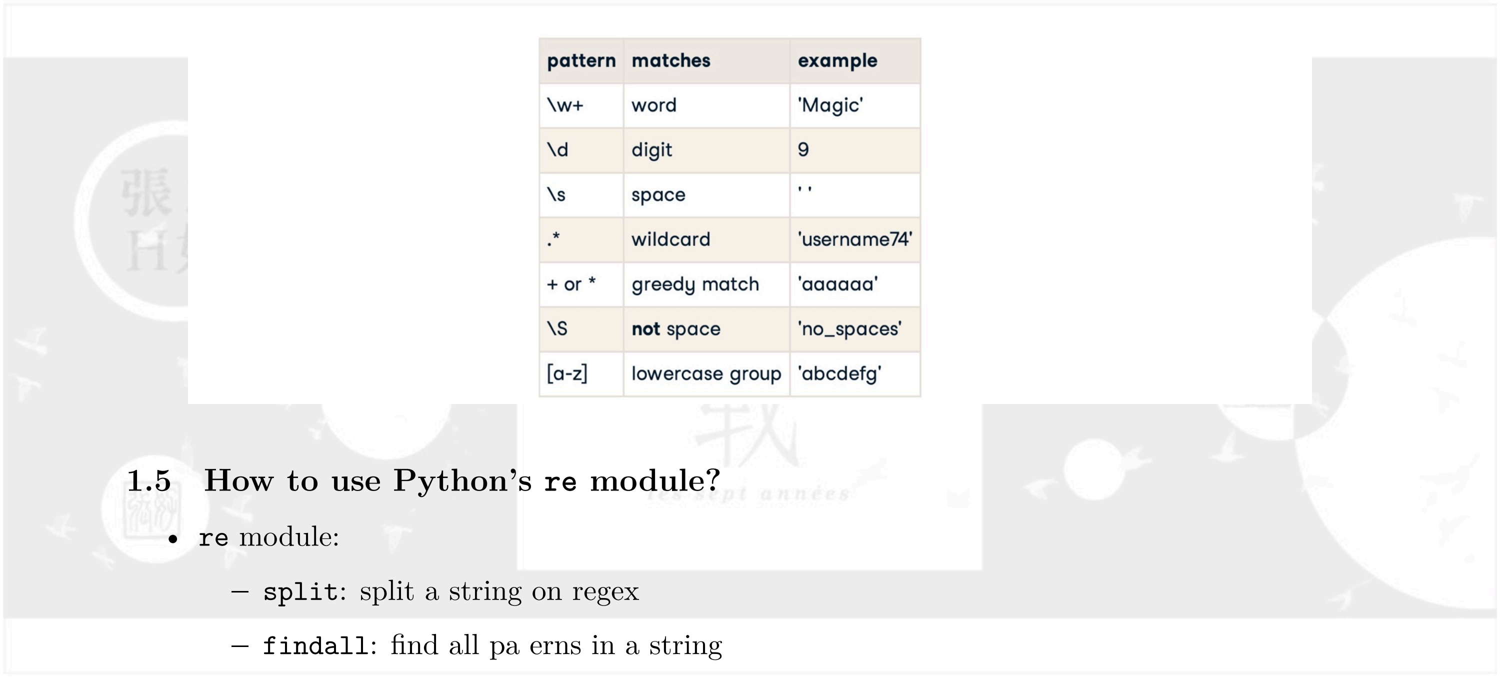
```
[1]: import re

     re.match('abc', 'abcdef')
```

```
[1]: <re.Match object; span=(0, 3), match='abc'>
```

```
[2]: word_regex = '\w+'
     re.match(word_regex, 'hi there!')
```

```
[2]: <re.Match object; span=(0, 2), match='hi'>
```

1.4 What are the common regex patterns?



pattern	matches	example
\w+	word	'Magic'
\d	digit	9
\s	space	' '
.*	wildcard	'username74'
+ or *	greedy match	'aaaaaa'
\S	**not** space	'no_spaces'
[a-z]	lowercase group	'abcdefg'

1.5 How to use Python's `re` module?

- `re` module:
  - `split`: split a string on regex
  - `findall`: find all pa erns in a string
  - `search`: search for a pattern
  - `match`: match an entire string or substring based on a pattern
- Parameterize the pattern first and parameterize the string second.
- May return an iterator, string, or match object.

1.6 Code of Python's `re` module:

```
[3]: re.split('\s+', 'Split on spaces.')
```

```
[3]: ['Split', 'on', 'spaces.']
```

1.7 Practice question for finding out the corresponding pattern:

- Which of the following regex patterns results in the following text?

```
>>> my_string = "Let's write RegEx!"
>>> re.findall(PATTERN, my_string)
['Let', 's', 'write', 'RegEx']
```

☐ `PATTERN = r"\s+"`.

☒ `PATTERN = r"\w+"`.

☐ `PATTERN = r"[a-z]"`.

☐ `PATTERN = r"\w"`.

▶ **Package pre-loading:**

[4]:
```
import re
```

▶ **Data pre-loading:**

[5]:
```
my_string = "Let's write RegEx!"
```

▶ **Question-solving method:**

[6]:
```
PATTERN = r"\s+"
re.findall(PATTERN, my_string)
```

[6]:
```
[' ', ' ']
```

[7]:
```
PATTERN = r"\w+"
re.findall(PATTERN, my_string)
```

[7]:
```
['Let', 's', 'write', 'RegEx']
```

[8]:
```
PATTERN = r"[a-z]"
re.findall(PATTERN, my_string)
```

[8]:
```
['e', 't', 's', 'w', 'r', 'i', 't', 'e', 'e', 'g', 'x']
```

[9]:
```
PATTERN = r"\w"
re.findall(PATTERN, my_string)
```

[9]:
```
['L', 'e', 't', 's', 'w', 'r', 'i', 't', 'e', 'R', 'e', 'g', 'E', 'x']
```

1.8 Practice exercises for introduction to regular expressions:

▶ **Package pre-loading:**

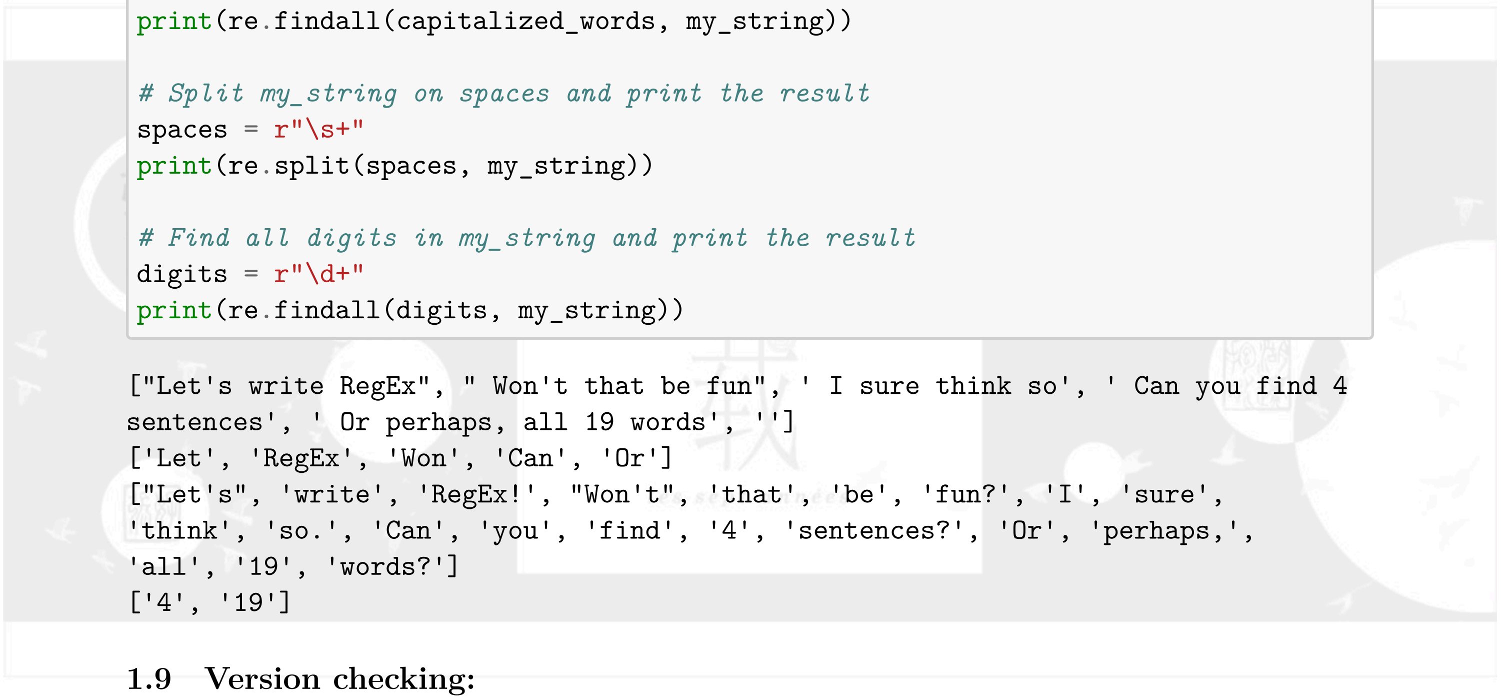
[10]:
```
import re
```

▶ **Data pre-loading:**

[11]:
```
my_string = "Let's write RegEx! Won't that be fun? I sure think so. \
Can you find 4 sentences? Or perhaps, all 19 words?"
```

▶ **Regular expressions (`re.split()` and `re.findall()`) practice:**

[12]:
```
# Write a pattern to match sentence endings: sentence_endings
sentence_endings = r"[\.\?!]"

# Split my_string on sentence endings and print the result
print(re.split(sentence_endings, my_string))

# Find all capitalized words in my_string and print the result
capitalized_words = r"[A-Z]\w+"
print(re.findall(capitalized_words, my_string))

# Split my_string on spaces and print the result
spaces = r"\s+"
print(re.split(spaces, my_string))

# Find all digits in my_string and print the result
digits = r"\d+"
print(re.findall(digits, my_string))
```



```
["Let's write RegEx", " Won't that be fun", ' I sure think so', ' Can you find 4
sentences', ' Or perhaps, all 19 words', '']
['Let', 'RegEx', 'Won', 'Can', 'Or']
["Let's", 'write', 'RegEx!', "Won't", 'that', 'be', 'fun?', 'I', 'sure',
'think', 'so.', 'Can', 'you', 'find', '4', 'sentences?', 'Or', 'perhaps,',
'all', '19', 'words?']
['4', '19']
```

1.9 Version checking:

[13]:
```
import sys

print('The Python version is {}.'.format(sys.version.split()[0]))
```

```
The Python version is 3.7.9.
```