# TP1: Introduction to Natural Language Processing

Dr. BENDIABDALLAH Mohammed Hakim
University of Ain Temouchent
Department of Computer Science
Module: Natural Language Processing

2025

## 1 Introduction

Welcome to TP1 for the Natural Language Processing module. In this lab, you will learn basic text handling in Python. The exercises in this tutorial will guide you through:

- Writing and reading text from a file.
- Using regular expressions to search for patterns.
- Tokenizing text by splitting it into words.
- Extracting specific personal data (email, telephone, birth date) using regex.

Please type the commands exactly as shown in the code boxes below.

## 2 Writing to a File

Before reading the file, we first create one with some personal data. In this example, we write the following information into a file named `sample.txt`:

```
bendiabdallah hakim 01/01/1989 0777777777
hakim.bendiabdallah@univ-temouchent.edu.dz
```

### Step 0: Write Data to the File

Type the following commands in your Python interpreter or script:

```
[0]: with open("sample.txt", "w") as file:
         file.write("bendiabdallah hakim 01/01/1989 0777777777
                     hakim.bendiabdallah@univ-temouchent.edu.dz")
```

**Explanation:** This code creates (or overwrites) the file `sample.txt` with the sample data.

## 3 Reading from a File

Now, we will read the content of the file and display it.

### Step 1: Read the File

Type the following commands:

[1]:
```
with open("sample.txt", "r") as file:
    text = file.read()
print(text)
```

**Explanation:** This code opens `sample.txt` in read mode, stores its content in the variable `text`, and then prints it.

## 4 Tokenization

Tokenization means splitting the text into smaller units (tokens), such as words. Here, we split the text by spaces.

### Step 2: Tokenize the Text

Type the following command:

[2]:
```
tokens = text.split()
print(tokens)
```

**Explanation:** This command splits the variable `text` at every space and prints the resulting list of tokens.

## 5 Using Regular Expressions

Python's built-in `re` library allows you to search for specific patterns in text.

### 5.1 Example Simple

Before processing our file, here is a very simple example. Suppose we have the sentence: `"Le chat dort sur le tapis."` We want to extract the word `"chat"`.

Type the following:

[3]:
```
import re
simple_text = "Le chat dort sur le tapis."
simple_pattern = r"chat"
result = re.search(simple_pattern, simple_text)
print(result.group())
```

**Explanation:** This simple example uses the pattern `"chat"` to find and print the substring `"chat"` in the sentence.

### 5.2 Table of Regular Expression Symbols

Below is a table of common regular expression symbols and their definitions:

| Expression | Definition |
|---|---|
| `\d` | A digit (0-9) |
| `\D` | A non-digit |
| `\s` | A whitespace character |
| `\S` | A non-whitespace character |
| `\w` | An alphanumeric character (letters, digits, underscore) |
| `\W` | A non-alphanumeric character |
| `.` | Any character except newline |
| `^` | Start of the line |
| `$` | End of the line |

## 5.3 Extracting Personal Data with Regular Expressions

Now we will extract specific fields from the file content. Recall that the file `sample.txt` contains: `bendiabdallah hakim 01/01/1989 0777777777 hakim.bendiabdallah@univ-temouchent.edu.dz`

### Step 3a: Extract the Email

Type the following:

[3a]:
```
email_pattern = r"(\S+@\S+\.\S+)"
email_result = re.search(email_pattern, text)
if email_result:
    print("Email:", email_result.group(1))
else:
    print("No email found")
```

### Step 3b: Extract the Telephone Number

Assuming the telephone number is a sequence of 10 digits:

[3b]:
```
tel_pattern = r"\b\d{10}\b"
tel_result = re.search(tel_pattern, text)
if tel_result:
    print("Telephone:", tel_result.group())
else:
    print("No telephone found")
```

\b ensures that the 10-digit number is not part of a longer number or word. It must stand alone, separated by spaces or punctuation.

**Step 3c: Extract the Birth Date**

Assuming the birth date format is DD/MM/YYYY:

[3c]:
```
date_pattern =
date_result = re.search(date_pattern, text)
if date_result:
    print("Birth Date:", date_result.group())
else:
    print("No birth date found")
```

## 6 Summary and Exercises

In this TP, you have learned to:

1. Write data to and read text from a file.
2. Use the `re` library to search for patterns with regular expressions.
3. Tokenize text by splitting it into words.
4. Extract specific personal data (email, telephone, birth date) using regex.

### Exercises (Homework):

**Filtering Algerian Phone Numbers and University Emails:**

- Create a new file (for example, `algerian.txt`) that contains several records with Algerian telephone numbers and various email addresses. Among these records, include phone numbers for Mobilis and Ooredoo, and at least one email address from the University of Ain Temouchent (i.e., an address ending with `@univ-temouchent.edu.dz`).
- Write a Python program that reads this file and uses regular expressions to extract:
  - Only the Algerian telephone numbers corresponding to the mobile operators Mobilis and Ooredoo (for example, numbers starting with `07` for Mobilis and `05` for Ooredoo).
  - Only the email addresses belonging to the University of Ain Temouchent.

## 7 Conclusion

You have now completed TP1 on Natural Language Processing. Practice these commands and explore different patterns or tokenization techniques to deepen your understanding. If you have any questions, please ask!