1 Lesson 3 – Fundamentals of Python

1.1 Objectives (TO COMPLETE)

The goal of this group of hands-on exercises is to introduce participants to common tasks for managing programming environments. **In the first list of proposed exercises (LESSON_03_exercises_v1.0), the participants aimed to apply conditionals and loops**. In the following exercises, **the participants will have to apply knowledge to solve problems focused on loops, functions**. After these exercises, the participants are expected to be able to:

- Implement good practices used to maintain clear and organised source code.
- Solve problems applying conditional sentences with relational operators and logic operators.
- Solve problems **applying loop sentences, creation of functions**.

1.2 Code conventions

- ✓ To create the answer file for each proposed exercise, follow the following file naming convention. Let the file **l_03_02_exercise_01.py** be the answer to the first exercise proposed. The meaning of the proposed name is defined below as follows, **l_03 = lesson 03, 02=exercises second part, 01=exercise number 1**, which directly corresponds to the approach of the proposed exercise. Finally, after solving the problems, you will have a list of files as follows:
  - **l_03_02_exercise_01.py**
  - **l_03_02_exercise_02.py**

  …

  - **l_03_02_exercise_n.py**

  **** Where "n" is the last exercise*. ***
- ✓ All the exercises **must have a header indicating the author**, description and usage. Here and example (https://github.com/juancarlosmiranda/python_code_recipes/blob/main/workshop_aua_activities/HEADERS.txt)
- ✓ Put **comments in order to explain the sentences**.
- ✓ Name the variables and functions following the PEP8 convention. (https://peps.python.org/pep-0008/)
- ✓ Use the template proposed **"activity_01_01.py"**, to organise the code (https://docs.python.org/3/library/__main__.html?highlight=__init#idiomatic-usage).

1.3 Exercises

Some data structures are proposed below, on which the exercises are based.

Data	Definition in Python

1 16 15 5
9 2 6 14
10 7 3 13
8 11 12 4
	`data_matrix = [`
	`[1, 16, 15, 5],`
	`[9, 2, 6, 14],`
	`[10, 7, 3, 13],`
	`[8, 11, 12, 4]]`

Matrix 1)

Data	Definition in Python
seal elephant lion monkey dog cat	`animals_list = ['seal', 'elephant', 'lion', 'monkey', 'dog', '', 'empty']`

List 1)

1.3.1 Using loops

All the exercises cited below are based on the data structure Matrix 1). Examples of source code could be **"activity_04_01.py" and "activity_04_02.py"** for loop reference.

1. Create a program to **get the sum** on the main diagonal (1, 2, 3, 4). An extension of this program must be developed to calculate the sum of the second diagonal (5, 6, 7, 8).
2. Create a program to display the data stored in the first full row and first column (1, 16, 15, 5) and (1, 9, 10, 8) respectively.
3. Create a program to display the matrix values iterating element by element.
4. Create a program to iterate through the matrix, this program will filter all the values less than 10. Print the values filtered.
5. Using the List 1), create a program to copy element by element from "animals_list" into anoter list. The program will stop to copy when it found an element equal to " (empty is double quotes)

1.3.2 Using functions

Several examples of templates for functions are offered in **"activity_04_03.py".**

6. Create a function called **filter_matrix(),** with a matrix as parameter and a threshold. The function will return a filtered matrix. The function header proposed is cited below.
   `matrix_filtered = filter_matrix(matrix_data, threshold_value)`
7. Create a function to print element by element from a matrix received as parameter.

1.4 Final words, tips

x.

1.5 Recomended reading

x.