Tutored & Practical Work n°01

NOTIONS OF PROGRAMMING LANGUAGE PARADIGMS

Exercise 1 (First steps in Python)

In PyCharm, create a new project called PWs. Develop a program `PW_1_1` that follows the **imperative programming paradigm** that does the following:

- Create a list L1 and initialize it with 10 integers;
- Calculate the sum of the elements of L1;
- Display the content of this list and its sum.

Reminder:
To use a list in Python:

```
L = [1,2,3,4,5]
L.append(6)     # Adding an element to the table
x = L[i]        # Read element number 'i' from the table
```

Exercise 2

To represent a polynomial, we will use a list of integers. The list contains the coefficients and exponents of the polynomial.

In PyCharm, in the same PWs project, develop a program `PW_1_2` that follows the **imperative programming paradigm** that:

- Stores the degree and coefficients of a polynomial in an array that are entered from the keyboard;
- Evaluate the polynomial for a value of `x` entered from the keyboard.

Exercise 3

In PyCharm, in the same PWs project, develop a program `PW_1_3` that follows the **functional programming paradigm** that does the same work as in Exercise 1 (First steps in Python) using a function to calculate the sum of the elements of a list called `sumList(L)`.

Exercise 4

Write a program that follows the **functional programming paradigm** that solves the problem in Exercise 2. This program must contain the following three functions:

```
def input_polynomial(p)
def display_polynomial(p)
def eval_polynomial(p)
```

Exercise 5 (Classes and Objects)

In PyCharm, in the same PWs project, develop a program `PW_1_5` containing the class `point` with two properties `x, y` and the method `move()` to move the point by 5 pixels on both axes and `display()` to display the coordinates of this point.

Create an object (or instance) '`p`' of the class `point`, display its properties, move '`p`' twice, and finally display the '`p`' properties.

Reminder:

The properties of a class are defined in the constructor `__init__(self)` as follows:

```
class C:
    def __init__(self):          The constructor
        self.a = 0               'a' is a property of the class C initialized to 0
```

Exercise 6 (Static variables or class variables in Python)

Try to understand what the following two programs do, explain and give the results.

```
class A:
    x = 5

print(A.x)

my_a = A()
my_b = A()
print(my_a.x)
my_a.x = 10
print(my_a.x)
print(my_b.x)
```

```
class A:
    def __init__(self):
        self.x = 5

    @staticmethod
    def double(v):
        return 2*v

print(A.double(10))
```

Exercise 7 (Inputs and outputs in Python)

In PyCharm, in the same PWs project, develop a program `PW_1_7` containing the following two functions and test them.

- A function `display_file(pathFile)` that displays the content of a text file whose path and name are entered from the keyboard.
- A function `typewriter(outputPathFile)` to record what the user types on the keyboard line by line until he types the letter 'q' in a line.

Reminder:

- Output on screen: `print("Hello world!")`
- `input()`: This function reads a line from input, converts it to a string and returns that. Exp: `a = input("Tape a value:")`
  The type of 'a' is string.
  To cast 'a' to an integer: `int(a)`.
  Or directly `a = int(input("Tape a value:"))`
- To open a file in order to read its content:
  `myFile = open("C:/test.txt", 'r')`
- To read the content of the whole file:
  `s1 = myFile.read()`
- To read only one line from the file:
  `s2 = myFile.readline()`
- To open a file in order to write data to it:
  `myFile = open("C:/test.txt", 'w')`
- To write text to a file:
  `myFile.write("Hello world! \n")`
- To close a file:
  `myFile.close()`
- To delete a file:
  `Import os`
  `os.remove("C:/test.txt")`