Arrays

Exercise 1 — Waiting for holidays...

Please look carefully what happens when typing the instructions in Listing 1 and **take notes** (the use of a pen is not forbidden when interacting with a computer). A file that you can use can be downloaded from http://www.barsamian.am/2021-2022/S6ICTE/TP8_Arrays.py.

```
1  days = [ ]
2  print(type(days))
3  days = ["thursday", "friday", "saturday", "sunday", "sunday", "monday", "
       tuesday"]
4  print(len(days))
5  days = days + ["wednesday"]
6  print(days)
7  del(days[4])
8  print(days)
9  a = 13
10 while (a < 29):
11     a = a + 1
12     b = a % 7
13     print(days[b], a, "of october")
```

Listing 1: Arrays: first steps.

Exercise 2 — Average of an array (of numbers)

Given an array `a` of numbers, we want to output the average value of those numbers. Of course, `sum(a) / len(a)` would work (why?). But instead, I want you to avoid using the `sum` function. Write a function `array_average` that takes as input an array of numbers, and that outputs the average value of numbers contained in it.

- the call `array_average([32, 5, 12, 8, 3, 75, 2, 15])` must return `19`.

Remark: see our work on loops, http://www.barsamian.am/2021-2022/S6ICTE/TP5_Loops.pdf.

Exercise 3 — Minimum, maximum of an array (of numbers)

1. Write a function `array_minimum` that takes as input an array of numbers, and that outputs the minimum value in this array.
   - the call `array_minimum([32, 5, 12, 8, 3, 75, 2, 15])` must output `2`.
2. Write a function `array_maximum` that takes as input an array of numbers, and that prints the maximum value in this array.
   - the call `array_maximum([32, 5, 12, 8, 3, 75, 2, 15])` must output `75`.

The goal of this exercise is to use only loops, assignments and conditional statements, without using anything else provided by `Python`. In particular, I don't want you to use the functions `min` and `max` that would answer the question : in `Python`, `min([32, 5, 12, 8, 3, 75, 2, 15])` is precisely 2.

Exercise 4 — Evens and odds

Write a program that goes through all numbers in an array of integers and creates two new arrays. The first array will contain only the even numbers from the initial one, and the second array will contain only the odd numbers from the initial one.

- a call on the array `[32, 5, 12, 8, 3, 75, 2, 15]` must build two arrays:
  `evens` that will be equal to `[32, 12, 8, 2]`
  `odds` that will be equal to `[5, 3, 75, 15]`.

Exercise 5 — Polynomials

Let us consider the polynomial function $P(x) = 2 + 3 \times x + 4 \times x^2 + x^3 + 2 \times x^4$. Our goal is to implement an efficient function that takes as parameter a real number x and outputs $P(x)$.

1. Create a function $f1$ that makes this computation as it is written.

   How many multiplications are needed?

2. Create a second function $f2$ that first stores into different variables the values x^2, x^3 and x^4, and then evaluates $P(x)$, thanks to these variables.

   How many multiplications are needed this time?

3. Show that $P(x)$ can also be computed as $2 + x \times (3 + x \times (4 + x \times (1 + x \times 2)))$ — prove it using Geogebra, and also prove it with a hand-written computation. Create a third function $f3$ that evaluates $P(x)$ thanks to this formula.

   How many multiplications are needed this time? This method is called Horner's method.

We now want to make those computations for any polynomial function which has order 4 (in S5 you have seen quadratic formulas, which use order 2; here it is order 4 because the biggest power of x is x^4). Instead of using directly the values $2, 3, 4, 1, 2$ (the coefficients of the polynomial P) as in questions 1–3, we want to store the coefficients of the polynomial in an array `c`: for P, we have `c = [2, 3, 4, 1, 2]`.

4. Write a fourth function $f4$ that uses the values in the array `c` and makes the same computations as in question 3.

BONUS How would you extend this to work with polynomial functions of a bigger order?