Solving complex problems with Python

Objectives

Development of Python modules to solve complex problems

- Develop Python modules and classes
- Use test-driven development
- Learn how to work with exceptions
- Familiarize with special libraries e.g. numpy, matplotlib

Deadline

During **lab 8**: present the basic $\boldsymbol{MyVector}$ class (with getters and setters)

During **lab 9**: present extra features (defined in lab 9)

Beginning of **lab 10**: upload the whole solution

Requirements

1. Implement a solution for the following problem using classes and feature driven development
2. The solution should offer a console type interface that allows the user to input the data and visualize the output
3. Use only the standard and compound data types available in Python

The application should be developed along several iterations and the solution should ensure:

- Providing at least 10 data examples in the application
- Documentation and testing of each function (at least 3 assertions)
- Validation of data – when the user introduces invalid commands or data, a warning should be generated

Solve 3 from extra features in the 2nd iteration.
Use your registration number ($\boldsymbol{n_{reg}}$) to define the number of exercises you have to solve: $\boldsymbol{n_{reg}}\ \mathbf{mod}\ \mathbf{8}+\mathbf{9}$, $\boldsymbol{n_{reg}mod\ 4}+\ \mathbf{17}$, $\boldsymbol{n_{reg}mod\ 3}+\mathbf{21}$
e.g. my registration number is 1491
$1491\ mod\ 8+9=\ 3+9=12$
$1491\ mod\ 4+17=3+17=20$
$1491\ mod\ 3+21=0+21=21$
⇒ I have to solve exercises: **12, 20, 21**

Problem specification

A math teacher needs a program that helps students perform different vector operations.

1st. Iteration

A vector (class $\boldsymbol{MyVector}$) is identified by the following properties:

- $name\_id$ given as a string/int
- $colour$ given as one letter (possible values ‘r’, ‘g’, ‘b’, ‘y’ and ‘m’)
- $type$ given as a positive integer greater or equal to 1
- $values$ given as a list of numbers

The following features are offered by the program (to be implemented in class $\boldsymbol{MyVector}$):

1. Scalar operations:
    a. Add a scalar to a vector – add_scalar
    e.g. `[1,2,3] + 2 = [3,4,5]`
2. Vector operations:
    a. Add two vectors – add
    e.g. `[1,2,3] + [4,5,6] = [5,7,9]`
    b. Subtract two vectors – subtract
    e.g. `[1,2,3] - [4,5,5] = [-3,-3,-2]`
    c. Multiplication – $multiplication$
    e.g. `[1,2,3] * [4,5,5] = 29`
3. Reduction operations
    a. Sum of elements in a vector
    e.g. `for [1,2,3] sum is 6`
    b. Product of elements in a vector
    e.g. `for [1,2,3] product is 6`
    c. Average of elements in a vector
    e.g. `for [1,2,3] average is 2`
    d. Minimum of a vector
    e.g. `for [1,-2,3] minimum is -2`
    e. Maximum of a vector
    e.g. `for [1,2,-3] maximum is 2`

2nd. Iteration

The program manages several vectors (class $\boldsymbol{VectorRepository}$) and allows operations such as:

1. Add a vector to the repository
2. Get all vectors
3. Get a vector at a given index
4. Update a vector at a given index
5. Update a vector identified by $name\_id$
6. Delete a vector by index
7. Delete a vector by $name\_id$

8. Plot all vectors in a chart based on the $type$ and $colour$ of each vector (using library matplotlib). Type should be interpreted as follows: 1 – circle, 2 – square, 3 – triangle, any other value – diamond. (No tests needed for this function)

9. Get the sum of elements in all vectors.
10. Get the vector which represents the sum of all vectors.
11. Get the list of vectors having a given sum of elements.
12. Get the list of vectors having the minimum less than a given value.
13. Get the sum of all the elements in those vectors having a given color.
14. Get the max of all vectors having the sum greater than a given value.
15. Get the min of all vectors.
16. Get a list of values representing the multiplication of consecutive vectors in the repository.
17. Delete all vectors from the repository. 18. Delete all vectors for which the color is a given value.
18. Delete all vectors for which the product of elements is greater than a given value.
19. Delete all vectors that are between two given indexes.
20. Delete all vectors for which the max value is equal to a given value.
21. Update all vectors by adding a given scalar to each element.
22. Update the color of a vector identified by $name\_id$.
23. Update all vectors having a given type by setting their color to the same given value.

3rd. Iteration
Implement all features from iteration 1 using special libraries e.g. numpy

Submission

Total points: **10**

You need to submit an **archive** (e.g. .zip, .rar, etc) with the source code (**only** your own *.py* files created, without venv or other generated files) to the assignment on **Teams** before the deadline. Please use the following convention to name the archive file:

$sfmie1234\_A4.zip$, where s – first letter of your surname
f – first letter of your first name
mie – stand for mathematics informatics in English

1234 – is your registration number
$A4$ – number of the assignment

If something is not clear, please ask me.

Key

1p Default
1p Work during lab 8
1p Work during lab 9
4p All features correctly implemented
1p At least 10 data examples for each iteration
1p At least 3 assertions for each function
1p Documentation