Solving complex problems with Python

Objectives

Development of Python modules to solve complex problems

- Develop Python modules
- Work with standard and compound data types in Python
- Use test-driven development
- Familiarize with special libraries e.g. numpy

Deadlines

- ***Lab 6****:* features 1 and 2 *(work during the same lab)*
- ***Lab 7****:* feature 3 *(homework from Lab6)*
  feature 1 using special libraries *(work during the same lab)*
- ***Lab 8****:* features 2, 3 using special libraries *(homework from Lab7)*

Requirements

1. Implement a solution for one of the following problems based on 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 2 consecutive iterations as follows:

1. **Iteration 1**
   a. Implementation
      i. feature 1
      ii. feature 2
      iii. feature 3
   b. Use modular programming
   c. The solution should ensure:
      - Providing at least 10 data examples in the application
      - Documentation and testing of each function (at least 5 assertions)
      - Validation of data – when the user introduces invalid commands or data, a warning should be generated
2. **Iteration 2**
   a. Implementation using special libraries e.g. numpy
      i. feature 1
      ii. feature 2
      iii. feature 3

P1. Vectors

A **math teacher** needs a program that helps **students** perform different vector operations. The program manages several vectors and allows students to use the following features offered by the program:

1. **Scalar operations**
   a. Add a scalar to a vector
      `e.g. [1,2,3] + 2 = [3,4,5]`
2. **Vector operations**
   a. Add two vectors
      `e.g. [1,2,3] + [4,5,6] = [5,7,9]`
   b. Subtract two vectors
      `e.g. [1,2,3] - [4,5,5] = [-3,-3,-2]`
   c. 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`

P2. Vectors 2D

A **math teacher** needs a program that helps **students** perform different vector operations. The program manages several vectors and allows students to use the following features offered by the program:

1. **Scalar operations**
   a. Add a scalar to a vector
      `e.g. [[1,2,3],[10,12,23],[11,22,3]] + 2 =`
      `[[3,4,5],[12,14,25],[13,24,5]]`
2. **Vector operations**
   a. Add two vectors
      `e.g. [[1,2,3],[10,12,23]] + [[4,2,2],[1,2,2]] =`
      `[[5,4,5],[11,14,25]]`
   b. Subtract two vectors
      `e.g. [[1,2,3],[10,12,23]] - [[4,2,2],[1,2,2]] =`
      `[[-3,0,1],[9,10,21]]`
   c. Multiplication
      `e.g. [[1,2,3],[10,12,23]] * [[4,2],[1,2],[1,2]] =`
      `[[9,12],[75,90]]`

3. **Reduction operations**
   a. Sum of elements in a vector
      ```
      e.g. for [[1,2,3],[10,12,23]] sum is 51
      ```
   b. Product of elements in a vector
      ```
      e.g. for [[1,2,3],[0,12,23]] product is 0
      ```
   c. Average of elements in a vector
      ```
      e.g. for [[1,2,3],[10,12,26]] average is 9
      ```
   d. Minimum of a vector
      ```
      e.g. for [[1,2,3],[10,-12,23]] minimum is -12
      ```
   e. Maximum of a vector
      ```
      e.g. for [[1,2,3],[10,12,-23]] maximum is 12
      ```