Solving problems with Python

Objectives

Development of Python modules and functions

- Implement functions
- Learn how to separate code on modules which can communicate by calling the functions
- Work with standard and compound data types in Python
- Learn how to specify and test Python code
- Use Eclipse to develop Python applications

Deadlines

- ***Lab 3***: features 1 and 2 *(work during the same lab)*
  Upload your solution online before the end of Lab3.

- ***Lab 4***: features 3 and 4 *(homework from Lab3)*
  features 5 and 6 *(work during the same lab)*
  Upload your new solution online before the end of Lab4.

- ***Lab 5:*** deadline to finalize the entire application
  Upload your final solution online before the start of Lab5.

Requirements

- Implement the solution using feature driven development
- The solution should offer a console type interface that allows the user to input the data and visualize the output
- Use only the standard and compound data types available in Python

The application should be developed along 3 consecutive iterations as follows:

1. **Iteration 1**
   a. Implementation
      i. feature 1
      ii. feature 2
   b. Use procedural programming
   c. Give at least 10 data examples in the application (to facilitate testing)
   d. Each function should be documented and tested (at least 5 assertions)

2. **Iteration 2**
    a. Implementation
        i. feature 3
        ii. feature 4
    b. Use procedural programming
    c. Give at least 10 data examples in the application (to facilitate testing)
    d. Each function should be documented and tested (at least 5 assertions)

3. **Iteration 3**
    a. Implementation
        i. feature 5
        ii. feature 6
    b. Use modular programming (at least 2 modules: one for UI and one for the functions needed)
    c. Give at least 10 data examples in the application (to facilitate testing)
    d. Each function should be documented and tested (at least 5 assertions)

The application should allow the validation of data – when the user inputs invalid data or commands, the application should give a warning.

Problems

P1. Numeric arrays

A **math teacher** needs a program to help **students** test different number properties. The program manages an array of numbers and allows students to use the following features offered by the program:

1. **Add numbers in the array**
    - *add 23* – add 23 as the last element of the array
    - *insert 12 at 1* – insert number 12 at index 1; the index of the first element is 0
2. **Modify elements in the array**
    - *remove 1* – removes the element at index 1
    - *remove from 1 to 3* – removes the elements at indices 1, 2 and 3
    - *replace 1 3 5 with 5 3* – replaces all sub-arrays 1 3 5 with 5 3
3. **Print the numbers that have certain properties**
    - *prime from 1 to 5* – print the prime numbers from the array found at indices 1..5
    - *odd from 1 to 5* – print the odd numbers from the array found at indices 1..5
4. **Obtain different characteristics from sub-arrays**
    - *sum from 1 to 5* – print the sum of elements 1..5
    - *gcd from 1 to 5* – print the greatest common divisor of elements 1..5
    - *max from 1 to 5* – print the maximum of elements 1..5
5. **Filter values**
    - *filter prime* – keep only prime numbers, remove the other elements
    - *filter negative* – keep only negative numbers, remove the other elements
6. **Undo**
    - *undo* – undo the last operation that modified the array

P2. Programming competition

In a programming competition, after the evaluation of solutions, the **evaluation committee** records in an array the scores obtained by **participants** after solving the problems (at index **i** in the array, the score of the **i-th** participant is stored). Given that the participants to the competition had to solve 10 problems, each evaluated to a maximum of 10 points, help the committee to access the following features offered by the program:

1. **Add the result of a new participant to the array**
   - *add 98* – add score 98 for the last participant
   - *insert 74 at 5* – insert at index 5 the score 74; the index of the first element is 0
2. **Modify the scores in the array (as a result of appeals)**
   - *remove 1* – delete the score of participant at index 1
   - *remove from 1 to 3* – delete scores for participants at indices 1, 2 and 3
   - *replace 4 with 55* – replace the score of participant at index 4 with score 55
3. **Print the participants with scores having some properties**
   - *less than 40* – print the participants with scores less than 40
   - *sorted* – print all participants sorted by their score
   - *sorted and greater than 90* – print the participants with scores higher than 90 sorted
4. **Obtain different characteristics of participants**
   - *avg from 1 to 5* – print the average of scores for participants 1..5
   - *min from 1 to 5* – print the smallest score for participants 1..5.
   - *mul 10 from 1 to 5* – print the scores for participants 1..5 which are multiples of 10
5. **Filter the scores**
   - *filter mul 10* - keep only participants with scores multiple of 10 (those who solved all / some problems with the highest score of 10), removing the other participants (scores)
   - *filter greater than 70* – keep only participants with scores higher than 70, removing the other participants (scores)
6. **Undo**
   - *undo* – undo the last operation that modified the array