Programming for IoT applications
Lab 1

Exercise 1. Develop in Object Oriented Programming (OOP) a simple calculator. The program will display a menu asking end-user to insert the operation to be performed and the two operands. The output should be a JSON reporting the input operands, the executed command and the result.
The accepted commands are:

- **add**: to add the operands and print the JSON;
- **sub**: to subtract the operands and print the JSON;
- **mul**: to multiply the operands and print the JSON;
- **div**: to divide the operands and print the JSON. CHECK that the operation is possible, if not an exception must be raised;
- **exit**: to close the program.

Validate each output JSON with jsonlint (http://jsonlint.com/)

Example of commands:

```
add 12 4.6
sub 3 12
```

Exercise 2. Extend *Exercise_1* to develop an OOP calculator where each method receives a list of numerical values, instead of 2, and print the result. The output should be a JSON reporting the input operands, the executed command and the result.
Validate each output JSON with jsonlint (http://jsonlint.com/)

Example:
Given the list [1, 2, 4.5, 7], the result of the **add** command is 1 + 2 + 4.5 + 7

Exercise 3. Develop in OOP a program for managing a discography. The full list of albums is stored in a file in the following JSON format:

```
{
  "discography_owner": "Tony Stark",
  "last_update": "2015-10-13 18:15",
  "album_list": [{
      "artist": "Bob Marley & The Wailers",
      "title": "Rastaman Vibration",
      "publication_year": 1976,
      "total_tracks": 11
    },{
      "artist": "Pink Floyd",
      "title": "The Wall",
      "publication_year": 1979,
        "total_tracks": 30
    },{
        "artist": "The Clash",
        "title": "Sandinista!",
        "publication_year": 1980,
        "total_tracks": 36
    },{
        "artist": "The Clash ",
        "title": "London Calling",
        "publication_year": 1978,
        "total_tracks": 19
    }]
}
```

The program needs to load the file and manage the discography providing the following features:

- **search_by_artist <artist_name>**: print all the information about the discs for the given <artist_name>
- **search_by_title <title>**: print all the information about the discs for the given <title>
- **search_by_publication_year <year>**: print all the information about the discs for the given <year>
- **search_by_total_tracks <total_tracks>**: print all the information about the discs having the given <total_tracks>
- **insert <artist> <title> <publication_year> <total_tracks>:** insert a new disc if and only if this is not already present in the list. Otherwise ask the end-user to update the information about the existing disc with the new parameters. Every time that this operation is performed the "last_update" field needs to be updated with the current date and time in the format "yyyy-mm-dd hh:mm".
- **print_all:** print the full discography
- **exit:** save the discography (if changed) in the same JSON file provided as input.

Finally, once the update file has been saved, validate the new JSON with jsonlint (http://jsonlint.com/)