Software Construction (L+E) HS 2023

Instructor: Prof. Dr. Alberto Bacchelli — Assignment 1

Teaching Assistant: Konstantinos Kitsios — Week 03

To correctly complete this assignment you **must**:

- Carry out the assignment with your team only (unless otherwise stated). You are allowed to discuss solutions with other teams, but each team should come up its own personal solution. A strict plagiarism policy is going to be applied to all the artifacts submitted for evaluation
- Prepare the solutions to the exercises by strictly following this structure:
  - A root folder named: `Group[id on OLAT]-a[AssignmentNumber]`[1]
  - Inside the folder:
    - a `README.md` file that explains the decisions taken in the source code and documents your solution;
    - the `file_manager.py` file that we provide you and is explained in Section 2;
    - a `run_tests.py` file that you have to write and will contain tests for the functions included in the provided `file_manager.py` and is explained in Section 3.
- The final structure of the directory should be as in Figure 1
- Package your root folder into a single ZIP file named: `Group[id on OLAT]-a[AssignmentNumber].zip`[2]
- Upload the solution to the right OLAT task by the deadline (*i.e.*, **Oct 24, 2023 @ 18:00**)

Figure 1: Directory Structure

```
Group1-a1/
├── file_manager.py
├── run_tests.py
└── README.md
```

---

1 Forming Groups

The assignment should be submitted by groups of *three* students. Find all the group members, then submit your names and emails by using this form: `https://forms.office.com/e/1zMNmR050j`.

2 file_manager.py

We have implemented some basic file manipulation functions in the `file_manager.py`, namely:

1. `read_file(file_name)` that returns the content of the file if the file exists, `None` otherwise;
2. `create_file(file_name, content="")` that returns `True` if the file was successfully created, `False` otherwise;

[1] *e.g.*, a correct name would be: `Group1-a1`

[2] *e.g.*, a correct name would be: `Group1-a1.zip`.

3. `write_file(file_name, content)` that returns `True` if the content was successfully written to the file, `False` otherwise;

4. `delete_file(file_name)` that returns `True` if the file was successfully deleted, `False` otherwise.

In the following section you will be asked to write tests that ensure these functions are working properly.

3 run_tests.py

Your task is to write a `run_tests.py` file containing test functions for each one of the four functions described above that will ensure that they are working as intended and described.

For example, to test that the function `read_file` works as described you could call it with a file that exists as input and assert it returns its contents. Another test would be to call it with a file that does not exist as input and assert it returns `None`.

The `run_tests.py` file should be implemented according to the guidelines discussed in the book and the lectures:

- implement three possible states for a test: `pass`, `fail` or `error`.
- implement *introspection*, i.e., the program should find and execute tests automatically.

.

Additionally to the above standard functionalities, you should also implement the following:

- Print the result of each test and how much time its execution took.
- Implement `setup` and `teardown` functions. The `setup` function should run before the execution of **each test** to setup the environment.[3] The `teardown` function should run after the execution of **each test** to cleanup the environment [4].
- Provide the user with an optional parameter `--select pattern` that, when specified, runs only the tests that contain the `pattern` in their name. For example, the command-line call

  ```
  run_tests --select read
  ```

  should run only the tests that contain the string `read` in their name.

4 Grading Criteria

Your assignment will be graded based on the following criteria:

- Creation of meaningful test functions for the four provided functions.
- Correct implementation and use of `setup` and `teardown` functions. [5]
- Generation of detailed test reports.
- Thoroughly documenting your decisions in the `README.md` file.

Assignments will be scored based on two main factors: fulfillment of criteria and evidence of effort:

- **2 out of 2:** For assignments that meet all criteria and clearly demonstrate substantial effort.
- **1 out of 2:** For assignments that meet some criteria but still show evident effort.
- **0 out of 2:** For assignments that lack clear evidence of effort, regardless of criteria met.

[3] For example, this is a good place to create files you want to use in your tests. You should not use the code we provided for creating a new file. Can you explain why?

[4] For example, this is a good place to delete all files that were used by the test and are not useful anymore

[5] For example, forgetting to delete files generated for testing is considered a very bad practice