January 7, 2025

Python Advanced Course

List 12.

The following instructions apply to tasks performed on lists 11–12.

(A) Using dedicated pyunit or nose environments, prepare the sets
unit tests to check the correctness of previously programmed tasks. Requirements:

- tests are located in a separate directory, outside the source code directories
  in the server or user interface;
- the tests are combined into a test set so that they can be used in one
  run all tests on the fly;
- prepare at least two classes with test cases, at least
  less three tests in each class.

(B) Look for information about PEP 8 (Python Enhancement Proposals). Using
pep8 checker or pycodestyle[1] packages check the compliance of the source code
of tasks (and test implementations) with PEP 8 recommendations. By sending files
on SKOS put files with corrected source code formatting. In
README.md file, include information about which tool you used. You can
You can also place a Makefile file with a code check command instead.

(C) Look for information on how to automatically generate documentation based on the
source code and its comments. Generate such
documentation for both tasks in some popular format (html, pdf,
etc.). Instead of posting the documentation on SKOS, it is enough to post
file with the appropriate commands, e.g. Makefile or README.md.

To execute it is necessary to complete the comments; it is required to comment
all classes and functions (along with parameters). Local functions
or irrelevant methods do not need to be commented on.

(D) Supplement the source code with typical annotations and check its correctness. As
in the previous points, include information about the analyzer used in the Makefile
or README.md file.

The task is worth 4 points.

Marcin Mÿotkowski

[1] Plug-ins in the VS Code environment or online services also check compliance with PEP 8.