Loosely adapted from https://cs50.harvard.edu/python/2022/project/

Final Project (November 2024)

Once you have solved each of the course's problem sets, it's time to implement your final project, a Python program of your very own! The design and implementation of your project is entirely up to you, albeit subject to these requirements:

1. Your project must be implemented in Python.
2. Your project must have a main function and three or more additional functions. At least three of those additional functions must be accompanied by tests that can be executed with pytest.
3. Your main function must be in a file called **project.py**, which should be in the "root" (i.e., top-level folder) of your project.
4. Your 3 required custom functions other than main must also be in project.py and defined at the same indentation level as main (i.e., not nested under any classes or functions).
5. Your test functions must be in a file called **test_project.py**, which should also be in the "root" of your project. Be sure they have the same name as your custom functions, prepended with test_ (test_custom_function, for example, where custom_function is a function you've implemented in project.py).
6. You should implement at least one class, which should have at least one user defined method.
7. You are welcome to implement additional classes and functions as you see fit beyond the minimum requirement.
8. Implementing your project should entail more time and effort than is required by each of the course's problem sets.
9. Avoid or limit manual inputs from the user: inputs should be in general included in the code, or read from files.
10. Any pip-installable libraries that your project requires must be listed, one per line, in a file called **requirements.txt** in the root of your project.

Example of Project Structures

project.py

```
def main():
    ...
def function_1():
    ...
def function_2():
    ...
def function_n():
    ...
if __name__ == "__main__":
    main()
```

test_project.py

```
def test_function_1():
    ...
```

```
def test_function_2():
    ...
def test_function_n():
    ...
```

Example of file requirements.txt:

```
pandas
numpy
re
csv
sys
...
```

You are welcome, but not required, to collaborate with one or two classmates on your project. But a two- or three-person should entail twice or thrice the time and effort required by a one-person project.

Getting Started

Creating an entire project may seem daunting. Here are some questions that you should think about as you start:

What will your software do? What features will it have? How will it be executed?

What new skills will you need to acquire? What topics will you need to research?

If working with one or two classmates, who will do what?

In the world of software, most everything takes longer to implement than you expect. And so it's not uncommon to accomplish less in a fixed amount of time than you hope. What might you consider to be a good outcome for your project? A better outcome? The best outcome?

Consider making goal milestones to keep you on track.

How to Submit

You must complete all three steps!

Step 1 of 3 (for the group)

Create a GitHub (public) repository for your project (<u>one single repository for the group</u>) . Create a README.md text file that explains your project. This file should include your Project title, and a description of your project.

When any user clones the repository and executes project.py in a local machine, after pip-installing the packages listed in requirements.txt, the execution should run with no errors and produce the expected output(s).

Your README.md file should be minimally multiple paragraphs in length, and should explain what your project is, what each of the files you wrote for the project contains and does, and if you debated certain design choices, explaining why you made them. Ensure you allocate sufficient time and energy to writing a README.md that documents your project thoroughly.

Step 2 of 3 (individual)

Create an individual **short video** (that's no more than 5 minutes in length) in which you present your project to the world, as with slides, screenshots, voiceover, and/or live action. Your video should somehow include your project's title, your name, and any other details that you'd like to convey to viewers

Step 3 of 3 (individual)

Submit in Fenix the URL of your project GitHub repository and your video.

Evaluation guidelines

There will be a discussion with the instructor where all elements of the group should be present. The grade depends on the group project but is individual (i.e. distinct element of the group might get distinct grades).

Evaluation criteria

- Clarity of the report and video. Low: if the report is poor and doesn't explain clearly what the data and problem are. High: if data and problem are well described, if the approach to solve the problem is carefully justified, and if the video presentation is engaging.
- Relevance and novelty of the project. Low: if it is very similar to what was done in class or too simple. High: if the project is original, compelling and more complex than the problems discussed in class.
- Technical quality. Low: if the code to solve the problem is unnecessarily complex or if it contains errors. High: if it's correct and uses judiciously the techniques discussed in class to solve the problem at hand.
- Code organization (modularity, comments). Low: if the script is long, involved and difficult to follow. High: if there is a compact and clear main() function that establishes the main steps of the resolution, and calls several additional functions, each one commented and organized in a way that the code is clear and easy to follow and modify if needed.
- Testing. Low: if the test functions do not cover obvious interesting 'corner' cases. High: if test functions are sound and reliable.

To get a 'pass', the following requirements are necessary:

- After cloning the repository and installing the required packages, 'project.py' should be executed with no execution errors, creating the expected output(s)
- The project needs to be different from what was done in class, but can be a variation on a problem already discussed in class, with some changes in the scripts, inputs and outputs. This should be described in the report, where the changes with respect to what was done in class should be discussed.
- You should implement at least one class, which should have at least one user defined method.
- Test functions need to be included in the project.