e-Portfolio Activity: Improving Code Quality. Refer to the Mertz (2019) resource. Use some Python code which you have developed in the past and apply at least 3 of the strategies presented at the source to improve its quality. You can use the Jupyter Notebook workspace and save your work to your GitHub repository.

Response:

I selected my very first script produced during Launching in Computer Science (LCS) module, specifically a script that supports in managing a food item cupboard. It allows entry, searching and removal of entries inputted when prompted.

The initial code was built using the PyCharm IDE which automatically identifies issues with the code such as missing variables, unfilled parameters and long single lines of code, amongst other issues. This helped resolve the immediate portion of any code issues.

Mertz (2019) recommends to:

1. Structure your code well
2. Follow a style guide
3. Document everything

The code before the changes is placed in *before* directory, and after changes in *after* directory.

On structure

Currently the project is structured inefficiently. It could potentially even be described as “spaghetti code” (GeeksForGeeks, 2024) as there are currently long blocks of code that fulfil both testing and execution of the functionality, making it difficult to read. Additionally, all code is currently structured in a single *main.py* file.

Based on given recommendations and that I anticipate the project will expand, I have created directories for *project*, *tests* and *docs*. Within project, I have created separate scripts for the classes defined in the *main.py* file and imported the classes in main.

On style

As discussed, PyCharm does some checks, however more checks could be run. I have installed black locally on my system, as I have used it in as part of my work client project. Upon running Black on all Python scripts, it reformatted the code in multiple places, particularly tidying up and making the code more readable by adding newline characters around required conditional statements, formatted *print()* statements, user input statements and elsewhere.

On documentation

For any missing docstrings, I ran the pydocstyle library script introduced in one of the activities for the ePortfolio. Upon running the script, it identified I:

1. Had missing docstrings to describe the overall modules
2. Had incorrectly formatted existing docstrings, particularly multi-line docstring quotes should be on a separate line, among other issues
3. Had used non-imperative language in docstrings

The library helped me resolve all of the found issues by pointing to specific lines where this was happening.

References:

GeeksForGeeks (2024) Spaghetti Code. GeeksForGeeks. Available from: https://www.geeksforgeeks.org/spaghetti-code/ [Accessed October 2 2024]

Mertz, J. (2019). Writing Clean and Pythonic Code. Available at: https://dataverse.jpl.nasa.gov/file.xhtml?fileId=63890&version=1.0 [Accessed October 2 2024]