**e-Portfolio Activity: Improving Code Quality**

**Introduction**

In this activity, I reflect on the principles of writing clean and Pythonic code as presented by James Mertz in 2019, and I apply these principles to improve the quality of a Python code snippet from a past project.

**Original Code Snippet**

Here is the original code snippet that I have developed in the past. The purpose of this code is to determine if a given number, is a prime number.

A computer screen with colorful text

Description automatically generated

**Applied Strategies for Improvement**

**Strategy 1: Use Tools to Help You**

I utilized the following tools to improve the code:

* **Black**: An auto-formatting tool to ensure the code adheres to PEP 8 style guidelines.
* **PyLint**: A linting tool to identify code smells and non-adherent code style.

**Strategy 2: Follow a Style Guide**

Adherence to PEP 8 was enforced to improve readability and consistency. This included:

* Ensuring proper indentation and avoiding tabs.
* Limiting line lengths to enhance readability across various environments.

**Strategy 3: Document All the Things!**

Documentation was added to clarify the purpose and usage of the code. This included:

* A comprehensive docstring for the function.
* Inline comments where necessary to explain complex logic.

**Improved Code Snippet**

Here is the improved code with the applied strategies:

A computer screen shot of a number

Description automatically generated

**Conclusion**

By applying the strategies from Mertz (2019), the quality of the code has been significantly improved. The code is now more readable, better structured, and well-documented, which aligns with the Pythonic philosophy of clear and logical code for both small and large-scale projects.