Part 1 assignment (Research and review style guides or coding standards for Python):

Standards for variable names:

A variable name has to start with a letter or the underscore character

A variable name can't start with a number

A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)

Variable names are case-sensitive (age, Age and AGE are three different variables)

A variable name cannot be any of the Python keywords.

Variables and functions use snake_case meaning they are all lowercase with underscores

Constants use SCREAMING_SNAKE_CASE meaning it is all caps

Classes use PascalCase which is capitalized words

Private variables use snake_case but start with an underscore _snake_case

Standards for documenting functions:

Inside the function you want a multiline comment. The first line should be a brief summary of what the function does. Then with an enter space. where after you want the "Args:" with a list of the arguments for the function, their datatypes, and a brief description of what they are. After that you will want another enter space and to have the "Returns:" with a list of the returns for the function, their datatypes, and a brief description of what they are.

Part 2 assignment(1 page paper):

When coding, following the set standards for variable names and documenting functions is extremely important. Coding standards are not just about making things look neat. They are important for making your code readable and preventing future problems. Python has many set standards for variable names and function documentation.

When it comes to variables, there are multiple different rules to follow when giving them a name. There are rules all variables must follow, which are: a variable name has to start with a letter or the underscore character, a variable name can't start with a number, a variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _), variable names are case-sensitive (age, Age and AGE are three different variables), and a variable name cannot be any of the Python keywords. In addition to those rules, you must name your variables based on what they are. If you are naming regular variables and functions, you should use snake_case, meaning they are all lowercase with underscores. If you are naming a constant, you should use SCREAMING_SNAKE_CASE, meaning the variable should be written in all caps. If you are writing a class, you should use PascalCase, which is capitalized words. Lastly, if you are naming a private variable, you should use snake_case, but start with an underscore _snake_case. These are the naming rules you should follow for ideal code readability and functionality. By using these naming cases, it is easy to see what type of variable you are working with, whether it is a regular variable, a constant, or a private variable. These naming standards allow you to work smoothly, reduce errors, and increase the readability of your code, which is important when working in groups.

Another standard you can follow to help improve readibility of your code is function documentation. This helps possible group members or people looking at your code understand what a function does or is supposed to do, and can help make it easier to solve problems that may arise. In Python, a good standard for function documentation is to have a multiline comment inside the function, so people know what it is for and what information they need for the function. The first line in the comment should be a brief summary of what the function does. Then an enter space. Where, after you want the "Args:" with a list of the arguments for the function, their datatypes, and a brief description of what they are. After that, you will want another enter space and to have the "Returns:" with a list of the returns for the function, their datatypes, and a brief description of what they are. This is all you will need, and it will make it significantly easier for you or others to read and understand your code.

Overall, following these standard methods/rules will allow your code to run smoothly and prevent errors, compared to if you were not using them.

Sources for part 1 and 2:
https://en.wikiversity.org/wiki/Applied_Programming/Functions/Python3 https://peps.python.org/pep-0008/
https://www.w3schools.com/python/python_variables_names.asp