**Relational Graph Quality**:

To create a high-quality relational graph in Python using matplotlib, ensuring readable axis labels and clear relationships, you can adjust various parameters such as font size, figure size, and spacing. Below is an example code demonstrating how to create a clear and readable scatter plot with properly labeled axes:

**Categorical Graph Quality:**

To create a high-quality categorical graph in Python using matplotlib, ensuring readable axis labels and clear appearance, you can adjust various parameters such as font size, figure size, and spacing. Below is an example code demonstrating how to create a clear and readable bar chart with properly labeled axes:

**Statistical Graph Quality:**

To ensure high-quality statistical graphs in Python, such as heatmaps, confusion matrices, violin plots, box plots, or corner plots, it's important to adjust parameters like font size, figure size, and spacing to maintain readability and clarity. Below is a general approach to create such graphs with clear axes labels:

**\**

**Quality of Analysis**

Certainly! Below is an example of Python code demonstrating how to conduct a data analysis along with generating a graph using matplotlib library:

1. Consistent Naming: Variable and function names are descriptive and follow the snake\_case convention, which is consistent with PEP-8 recommendations.
2. Indentation: The code uses four spaces for indentation, which is the standard recommended by PEP-8.
3. Whitespace: Proper spacing is used around operators and after commas, following PEP-8 guidelines.
4. Line Length: Lines are kept within the recommended maximum line length of 79 characters, ensuring readability.
5. Comments: The code includes comments where necessary to explain the purpose of functions and provide context for specific actions. Comments are clear and concise.
6. Functionality: The code effectively achieves its purpose of visualizing data using a histogram and calculating basic descriptive statistics. It is well-organized and easy to understand.
7. Error Handling: While error handling is not explicitly implemented in the provided code, it may be beneficial to include error handling mechanisms to handle potential exceptions or edge cases.
8. Documentation: The code includes a docstring for the visualize\_data function, providing a brief description of its purpose, parameters, and return value. This enhances readability and makes the code more understandable for other developers.

**Submitted To**

**Dr. William Cooper**

(**module leader)**

**w.cooper@herts.ac.uk**

**Submitted by**

**Farid Hossain**

**Student Id: 23006446**

**Gmail Id: faridhossain7600@gmail.com**

**User Id: fh23abd@herts.ac.uk**

**Code Link/ Github id link:**