Final Project Programming Requirements

Each project, whether Provided or Custom, must include:

1. **At Least Two Python Functions**: Use docstrings to explain the purpose, parameters, and return values.

Example 1

```
def calculate_average(numbers):
    """
    Calculate the average of a list of numbers.

    Parameters
    ----------
    numbers : list
        A list of numerical values.

    Returns
    -------
    float
        The average of the numbers in the list.
    """
    return sum(numbers) / len(numbers)
```

2. **At Least Two Instances of Error Handling**: These could involve file I/O issues, invalid user input, or data-related errors (e.g., missing fields). Think `try/except` blocks.

Example 1

```
try:
    with open('data.txt', 'r') as file:
        data = file.read()
except FileNotFoundError:
    print("The file was not found.")
```

Example 2

```
try:
    user_input = int(input("Enter a number: "))
except ValueError:
    print("Invalid input. Please enter a valid number.")
```

3. **At Least One Loop**: This can include a `for`/`while` loop or a list comprehension.

Example 1

```
for i in range(10):
    print(i)
```

Example 2

```
numbers = [1, 2, 3, 4, 5]
for number in numbers:
    print(number)
```

Example 3

```
# list comprehension
squared_numbers = [x**2 for x in range(10)]
```

4. **At Least One Conditional**: An `if/else` or other branching logic.

Example 1

```
if x > 0:
    print("x is positive")
else:
    print("x is non-positive")
```

Example 2

```
number = int(input("Enter a number: "))
if number > 0:
    print("The number is positive.")
elif number < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

5. **At Least One Figure**: A graph, chart, or other plot generated by your Colab notebook - this is a note for those pursuing a Custom Projects. All Provided Projects will ask for visualization of the results.

Example 1

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]

plt.figure()

plt.plot(x, y)
plt.title('Simple Line Graph')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.show()
```