question4.py

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Load data from the provided CSV file.
data = pd.read_csv("HW2_linear_data.csv")
# Load the first (input/feature) column into the vector X.
X = data.iloc[:, 0].values
# Load the second (output/target) colulmn into the vector Y.
Y = data.iloc[:, 1].values

# Initialize the parameters to 0.
# The m value is the slope (i.e., weight), and the c value
# is the y-intercept (i.e., bias).
m = 0
c = 0

# Set learning rate and number of epochs as described in the problem statement.
learning_rate = 0.0001
epochs = 1000

# Count the number of input values.
n = len(X)

# Calculate the gradient descent.
for e in range(epochs):
    # Calculate the predicted value
    Y_pred = m * X + c
    # Calculate the error relative to ground truth.
    error = Y_pred - Y
    # Use the error value to calculate the MSE.
    mse = np.mean(error**2)

    # Update the slope/weight and intercept/bias based on the
    # learning rate and the error.

    # When updating m, the term term "np.sum(error * X)" represents
    # the gradient of the MSE with respect to m.
    m -= learning_rate * (2/n) * np.sum(error * X)  # Update slope
    # When updating c, the term "np.sum(error)" represents the
    # gradient of the MSE with resepct to c.
    c -= learning_rate * (2/n) * np.sum(error)  # Update intercept

    # Print the MSE value every 100 epochs to see changes.
    # The MSE at epoch 0 should be very large because of the initial
    # values of 0 for both m and c, with a sharp drop at epoch
    # epoch 100, followed by small incremental improvements for the
    # remaining epochs.
    if e % 100 == 0:
        print(f"Epoch {e}: MSE = {mse}")

# Generate the final predictions using the trained values of m and c.
Y_pred = m * X + c

# Plot the results.
# Generate a scatter plot of the ground-truth data.
plt.scatter(X, Y, color="blue", label="Actual Data")
# Plot the predictions as a line.
plt.plot(X, Y_pred, color="red", label="Fitted Line")
plt.xlabel("X")
plt.ylabel("Y")
plt.title("Linear Regression")
plt.show()

# Print the final trained parameters.
print(f"Final slope (m): {m}")
print(f"Final intercept (c): {c}")

```