ChatGPT Query

Write a python program to calculate and plot the first N entries in a series generated by the logistics equation.

ChatGPT Response

The logistic equation is a famous example of a nonlinear difference equation that exhibits chaotic behavior. It is given by:

$$x_{n+1} = rx_n(1 - x_n)$$

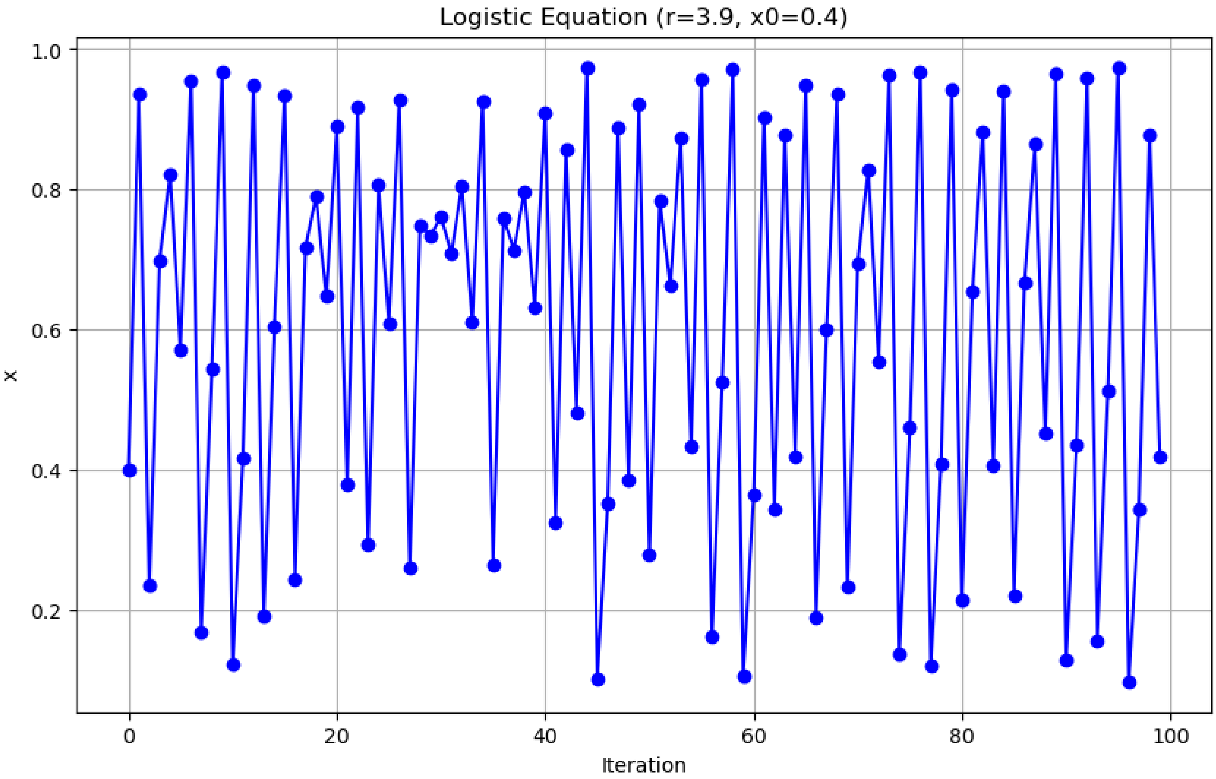
Where:

- x_n is the current value in the sequence.
- r is a parameter that controls the behavior of the sequence.

To calculate and plot the first N entries in a series generated by the logistic equation, you can use Python and the matplotlib library for plotting. Here's a Python program that does this:

In [3]:

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

def logistic_equation(r, x0, N):
    # Initialize lists to store the values of x and the iteration numbers
    x_values = []
    iteration_numbers = []

    x = x0  # Initial value
    for n in range(N):
        x_values.append(x)
        iteration_numbers.append(n)
        x = r * x * (1 - x)

    return iteration_numbers, x_values

# Parameters
r = 3.9  # You can change the value of r
x0 = 0.4  # Initial value
N = 100  # Number of iterations

# Calculate the logistic sequence
iteration_numbers, x_values = logistic_equation(r, x0, N)

# Plot the logistic sequence
plt.figure(figsize=(10, 6))
plt.plot(iteration_numbers, x_values, marker='o', linestyle='-', color='b')
plt.title(f'Logistic Equation (r={r}, x0={x0})')
plt.xlabel('Iteration')
plt.ylabel('x')
plt.grid(True)
plt.show()
```



In this program, you can change the values of r, x_0, and N to explore different behaviors of the logistic equation. The program calculates the values of x at each iteration and plots them. The resulting plot will show the evolution of the logistic sequence over the specified number of iterations.

End of ChatGPT Response

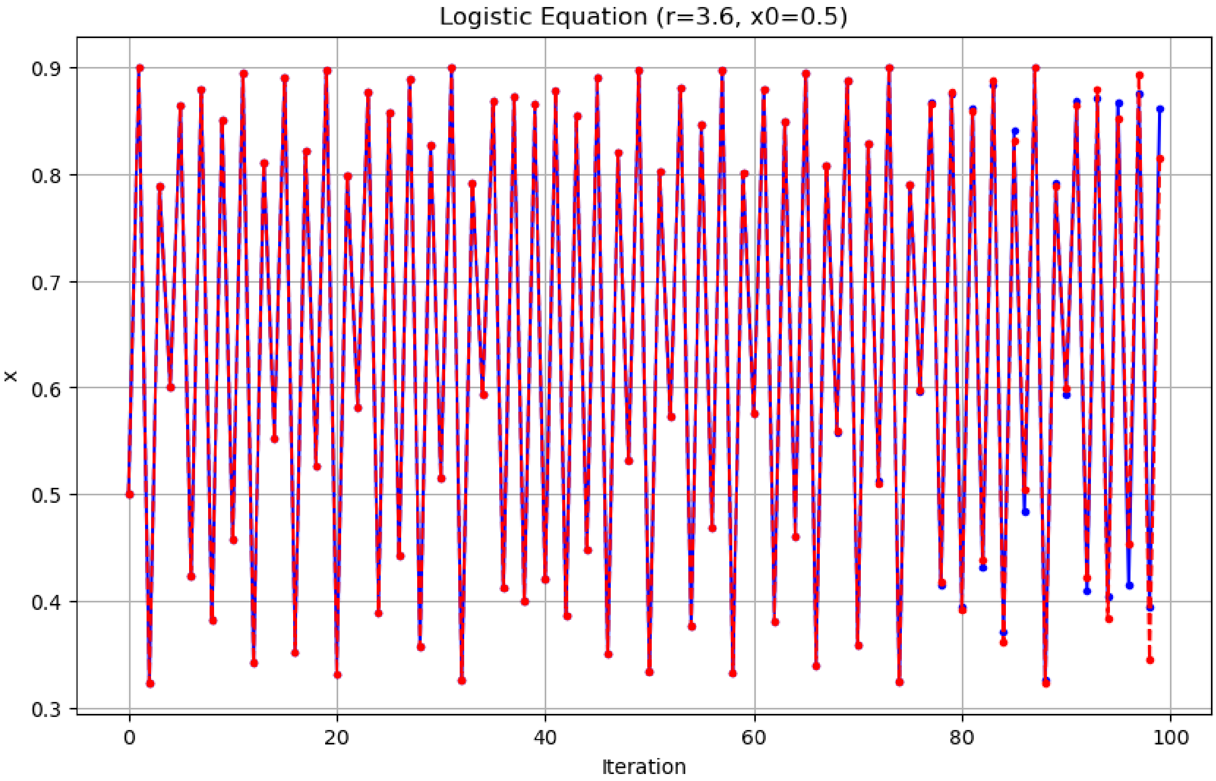
It worked right-out-of-the-box with modifications required!!

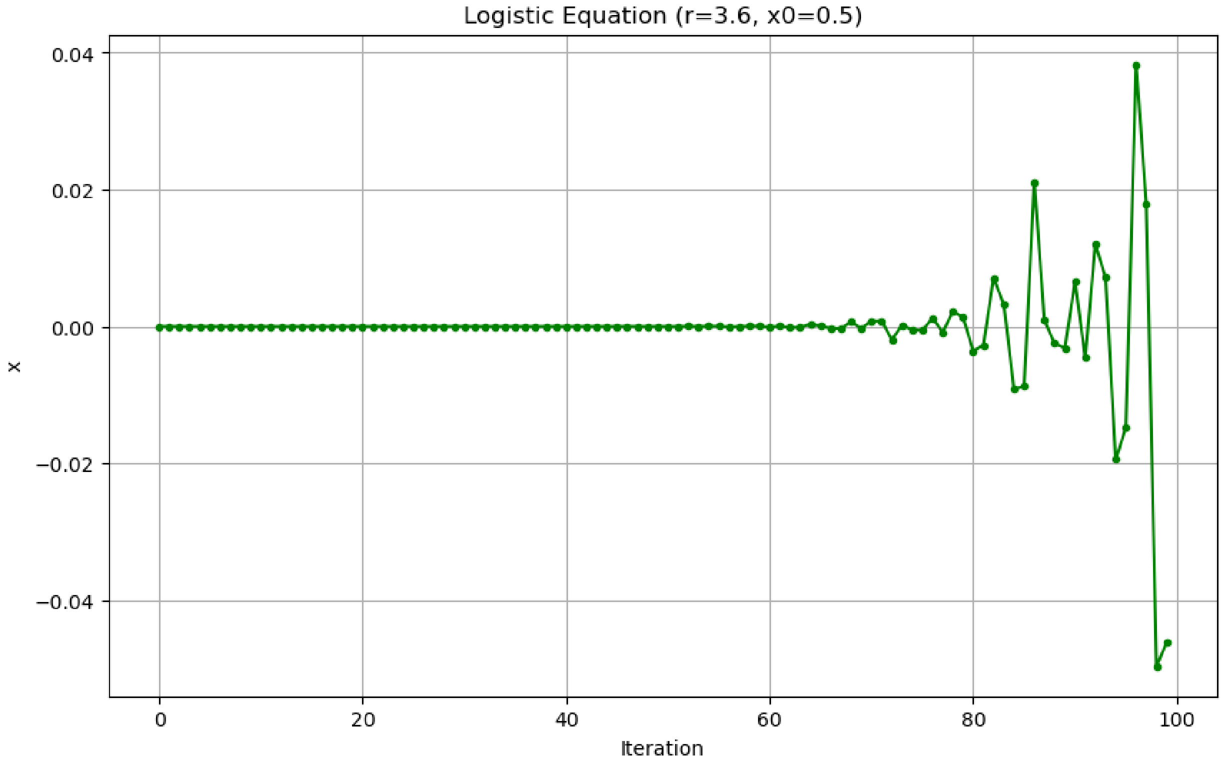
It labled the plots, AND included comments in the python program!!

In [2]:

```
# Parameters
r = 3.6  # You can change the value of r
x0 = 0.5  # Initial value
N = 100  # Number of iterations

# Calculate the logistic sequence
epsilon = 0.000005
iteration_numbers, x_true = logistic_equation(r, x0, N)
iteration_numbers, x_pred = logistic_equation(r, x0+epsilon, N)


# Plot the logistic sequence
plt.figure(figsize=(10, 6))
plt.plot(iteration_numbers, x_true, marker='.', linestyle='-', color='b')
plt.plot(iteration_numbers, x_pred, marker='.', linestyle='--', color='r')
plt.title(f'Logistic Equation (r={r}, x0={x0})')
plt.xlabel('Iteration')
```

```python
plt.ylabel('x')
plt.grid(True)
plt.show()

x_diff = []
for i in range(N):
    x_diff.append(x_pred[i]-x_true[i])

# Plot the logistic sequence
plt.figure(figsize=(10, 6))
plt.plot(iteration_numbers, x_diff, marker='.', linestyle='-', color='g')
plt.title(f'Logistic Equation (r={r}, x0={x0})')
plt.xlabel('Iteration')
plt.ylabel('x')
plt.grid(True)
plt.show()
```




Logistic Equation (r=3.6, x0=0.5)
0.04
0.02
0.00
−0.02
−0.04
x
0
20
40
60
80
100
Iteration

```
In [ ]:
```