In [154…

```python
#install Pandas
!pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\bridget\anaconda3\lib\site-packages
(1.2.4)
Requirement already satisfied: pytz>=2017.3 in c:\users\bridget\anaconda3\lib\site-packa
ges (from pandas) (2021.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\bridget\anaconda3\lib
\site-packages (from pandas) (2.8.1)
Requirement already satisfied: numpy>=1.16.5 in c:\users\bridget\anaconda3\lib\site-pack
ages (from pandas) (1.20.1)
Requirement already satisfied: six>=1.5 in c:\users\bridget\anaconda3\lib\site-packages
(from python-dateutil>=2.7.3->pandas) (1.15.0)
```

In [151…

```python
#import pandas
import pandas as pd
```

In [152…

```python
#Load Excel data
df = pd.read_excel('C:/Users/BRIDGET/Downloads/Eco710/chickegg.xlsx')
```

In [5]:

```python
#Lets see what the df data look like
df.head()
```

Out[5]:

	Year	chicken	egg	egg in million
0	1930	468491	3581.0	42972.0
1	1931	449743	3532.0	42384.0
2	1932	436815	3327.0	39924.0
3	1933	444523	3255.0	39060.0
4	1934	433937	3156.0	37872.0

In [6]:

```python
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\bridget\anaconda3\lib\site-package
s (3.3.4)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: python-dateutil>=2.1 in c:\users\bridget\anaconda3\lib\si
te-packages (from matplotlib) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\users\bridget\anaconda3\lib\site-packa
ges (from matplotlib) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\bridget\anaconda3\lib\site-
packages (from matplotlib) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\brid
get\anaconda3\lib\site-packages (from matplotlib) (2.4.7)
Requirement already satisfied: pillow>=6.2.0 in c:\users\bridget\anaconda3\lib\site-pack
ages (from matplotlib) (8.2.0)
Requirement already satisfied: numpy>=1.15 in c:\users\bridget\anaconda3\lib\site-packag
es (from matplotlib) (1.20.1)
Requirement already satisfied: six in c:\users\bridget\anaconda3\lib\site-packages (from
cycler>=0.10->matplotlib) (1.15.0)
```

In [7]:

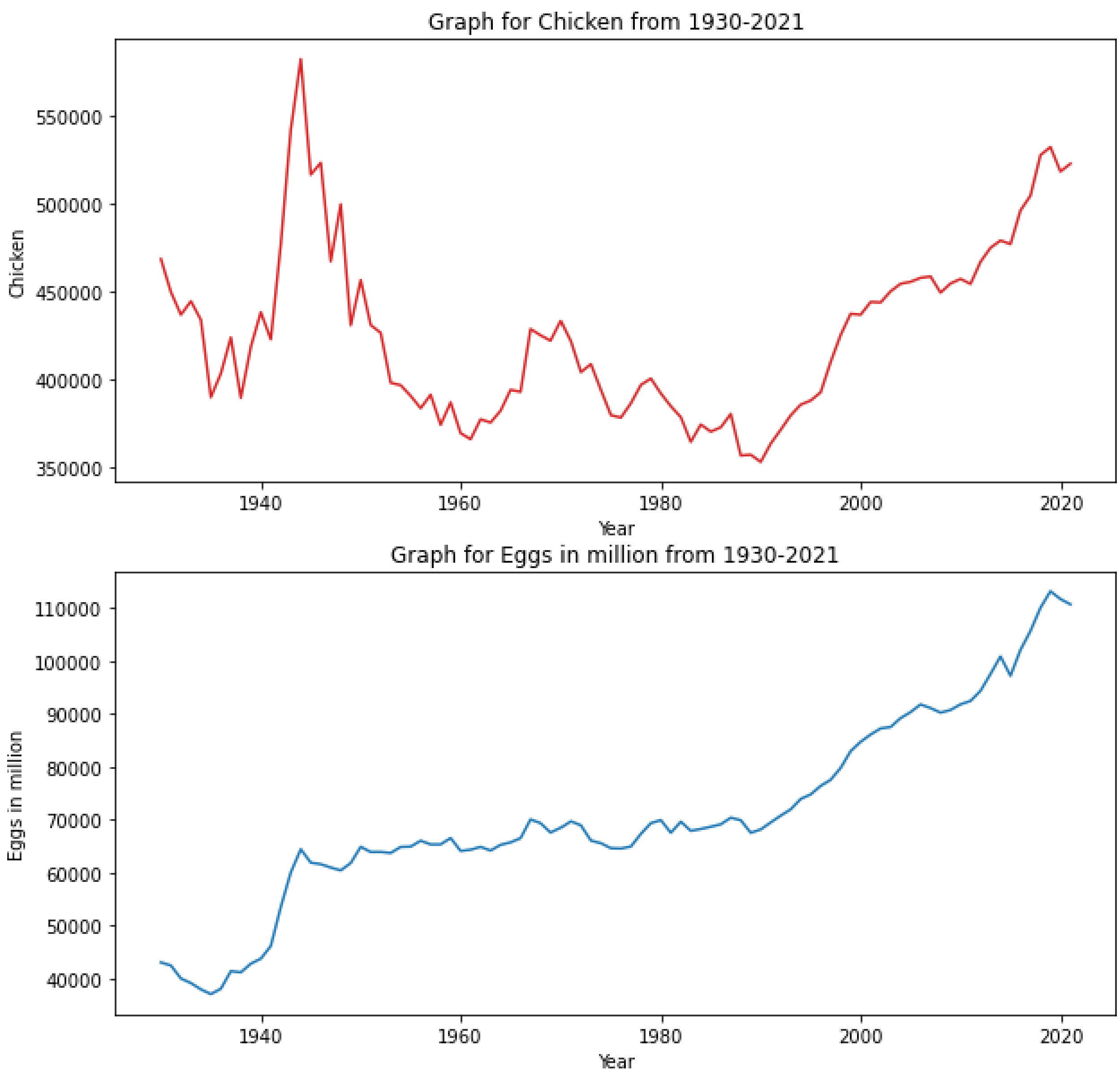
```
import matplotlib.pyplot as plt
```

In [28]:

```
fig, axs = plt.subplots(2, 1, figsize=(10, 10))
#graph for Chickens from 1930-2021
axs[0].plot(df['Year'], df['chicken'], 'tab:red')
axs[0].set_title('Graph for Chicken from 1930-2021')
axs[0].set_xlabel('Year')
axs[0].set_ylabel('Chicken')
#graph for Eggs from 1930-2021
axs[1].plot(df['Year'], df['egg in million'], 'tab:blue')
axs[1].set_title('Graph for Eggs in million from 1930-2021')
axs[1].set_xlabel('Year')
axs[1].set_ylabel('Eggs in million')
```

Out[28]:

```
Text(0, 0.5, 'Eggs in million')
```



In [26]:

```
#to adjust layout
plt.tight_layout()
```

<Figure size 432x288 with 0 Axes>

In [27]:

```
#to display the plot
plt.show()
```

In [217…

```
df3 = df[(df['Year'] >= 1930) & (df['Year'] <= 1983)].copy()
```

In [218…

```
df3.head(100)
```

Out[218…

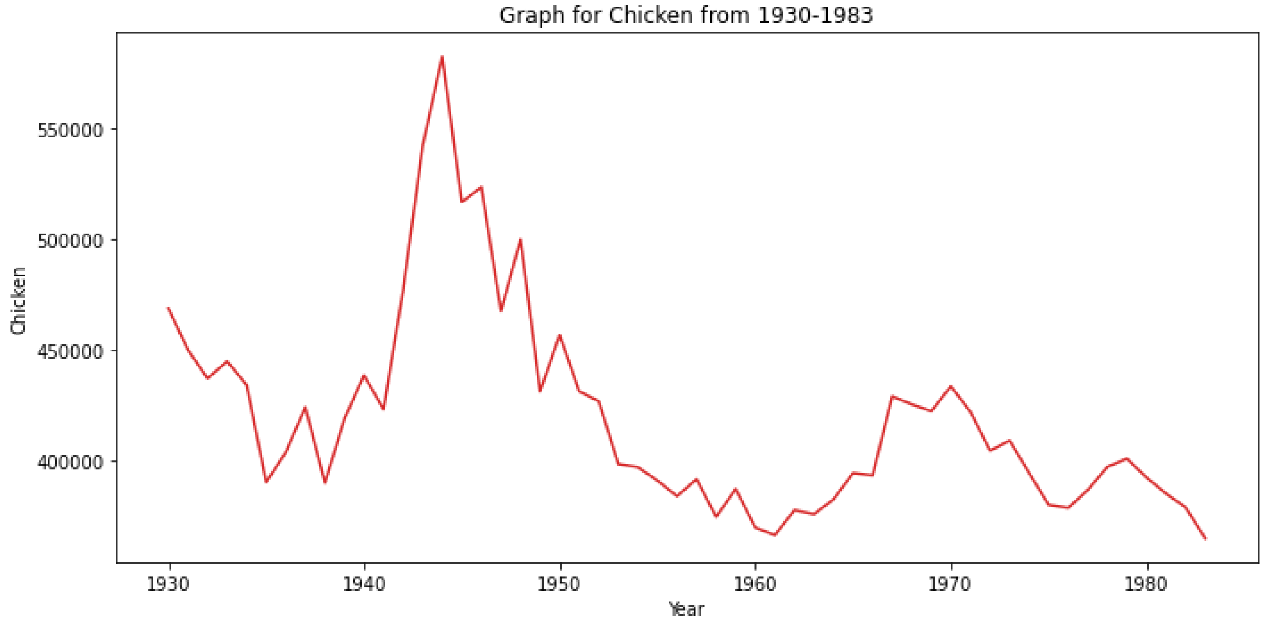
	Year	chicken	egg	egg in million
0	1930	468491	3581.0	42972.0
1	1931	449743	3532.0	42384.0
2	1932	436815	3327.0	39924.0
3	1933	444523	3255.0	39060.0
4	1934	433937	3156.0	37872.0
5	1935	389958	3081.0	36972.0
6	1936	403446	3166.0	37992.0
7	1937	423921	3443.0	41316.0
8	1938	389624	3424.0	41088.0
9	1939	418591	3561.0	42732.0
10	1940	438288	3640.0	43680.0
11	1941	422841	3840.0	46080.0
12	1942	476935	4456.0	53472.0
13	1943	542047	5000.0	60000.0
14	1944	582197	5366.0	64392.0
15	1945	516497	5154.0	61848.0
16	1946	523227	5130.0	61560.0
17	1947	467217	5077.0	60924.0
18	1948	499644	5032.0	60384.0
19	1949	430876	5148.0	61776.0
20	1950	456549	5404.0	64848.0
21	1951	430988	5322.0	63864.0
22	1952	426555	5323.0	63876.0
23	1953	398156	5307.0	63684.0
24	1954	396776	5402.0	64824.0
25	1955	390708	5407.0	64884.0

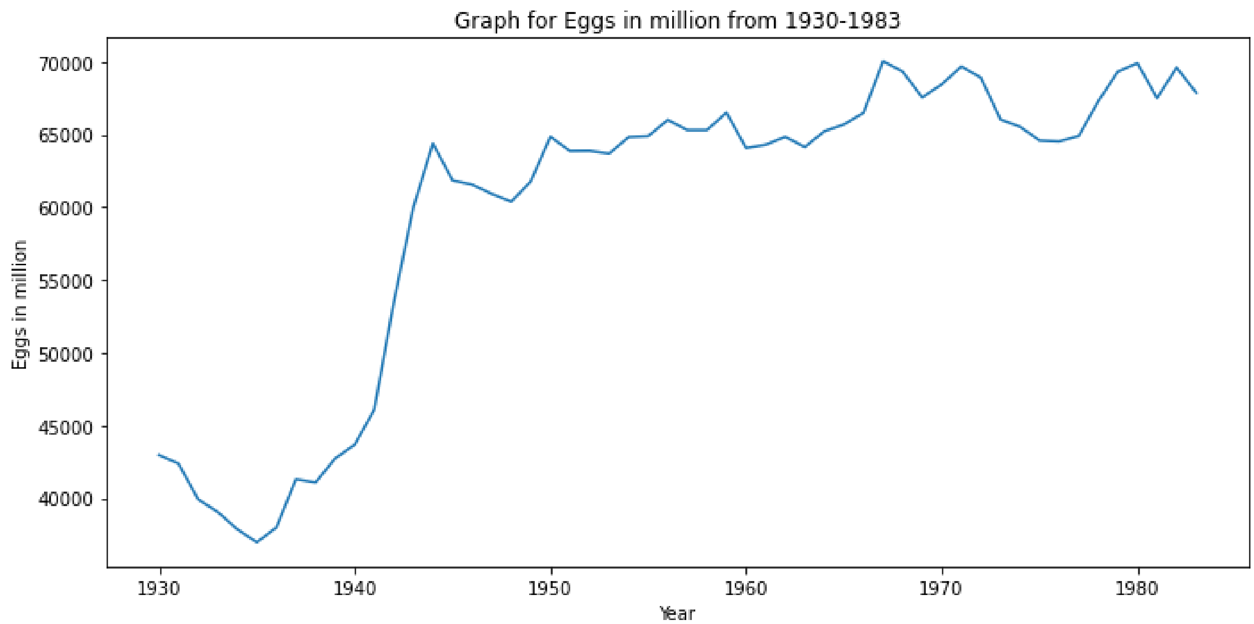
	Year	chicken	egg	egg in million
26	1956	383690	5500.0	66000.0
27	1957	391363	5442.0	65304.0
28	1958	374281	5442.0	65304.0
29	1959	387002	5542.0	66504.0
30	1960	369484	5339.0	64068.0
31	1961	366082	5358.0	64296.0
32	1962	377392	5403.0	64836.0
33	1963	375575	5345.0	64140.0
34	1964	382262	5435.0	65220.0
35	1965	394118	5474.0	65688.0
36	1966	393019	5540.0	66480.0
37	1967	428746	5836.0	70032.0
38	1968	425158	5777.0	69324.0
39	1969	422096	5629.0	67548.0
40	1970	433280	5704.0	68448.0
41	1971	421763	5806.0	69672.0
42	1972	404191	5742.0	68904.0
43	1973	408769	5502.0	66024.0
44	1974	394101	5461.0	65532.0
45	1975	379754	5382.0	64584.0
46	1976	378361	5377.0	64524.0
47	1977	386518	5408.0	64896.0
48	1978	396933	5608.0	67296.0
49	1979	400585	5777.0	69324.0
50	1980	392110	5825.0	69900.0
51	1981	384838	5625.0	67500.0
52	1982	378609	5800.0	69600.0
53	1983	364584	5656.0	67872.0

In [219…

```
fig, axs = plt.subplots(2, 1, figsize=(10, 10))
#graph for Chickens from 1930-1983
axs[0].plot(df3['Year'], df3['chicken'], 'tab:red')
axs[0].set_title('Graph for Chicken from 1930-1983')
axs[0].set_xlabel('Year')
axs[0].set_ylabel('Chicken')
#graph for Eggs from 1930-1983
axs[1].plot(df3['Year'], df3['egg in million'], 'tab:blue')
```

```
axs[1].set_title('Graph for Eggs in million from 1930-1983')
axs[1].set_xlabel('Year')
axs[1].set_ylabel('Eggs in million')
plt.tight_layout()
plt.show()
```





In [220…

```
from statsmodels.tsa.stattools import kpss, adfuller,grangercausalitytests
import numpy as np
```

In [221…

```
#Estimate the number of differencing required to make a time series stationary.
serieseggs = df3['egg in million']
serieschicken = df3['chicken']
```

In [222…

```
#Test for Stationarity using ADF test for egg in million
result = adfuller(df3['egg in million'])
result = adfuller(df3['egg in million'].dropna())
print('ADF Statistic: %f' % result[0])
```

```
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
print()
```

```
ADF Statistic: -1.715420
p-value: 0.423186
Critical Values:
	1%: -3.563
	5%: -2.919
	10%: -2.597
```

In [223…

```
#Interpretation
if result[1] > 0.05:
    print("egg in million is not stationary")
else:
    print("egg in million is stationary")
```

```
egg in million is not stationary
```

In [224…

```
#Test for Stationarity using ADF test for chicken
result = adfuller(df3['chicken'].dropna())
result = adfuller(df3['chicken'])
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
print()
```

```
ADF Statistic: -1.968629
p-value: 0.300495
Critical Values:
	1%: -3.566
	5%: -2.920
	10%: -2.598
```

In [225…

```
#Interpretation
if result[1] > 0.05:
    print("chicken is not stationary")
else:
    print("chicken is stationary")
```

```
chicken is not stationary
```

In [247…

```
# first order differencing for egg in million
df3['egg in million_diff'] = df3['egg in million'].diff().dropna()
# first order differencing for chicken
df3['chicken_diff'] = df3['chicken'].diff().dropna()
```

In [250…

```
df3['chicken_diff'] = df3['chicken_diff'].dropna()

# Check and drop NaN values
df3 = df3.dropna(subset=['chicken_diff'])
```

```python
df3 = df3.dropna(subset=['egg in million_diff'])
df3.reset_index(drop=True, inplace=True)

aligned_years = df3[df3['Year'].isin(df3.index)]
```

In [274…

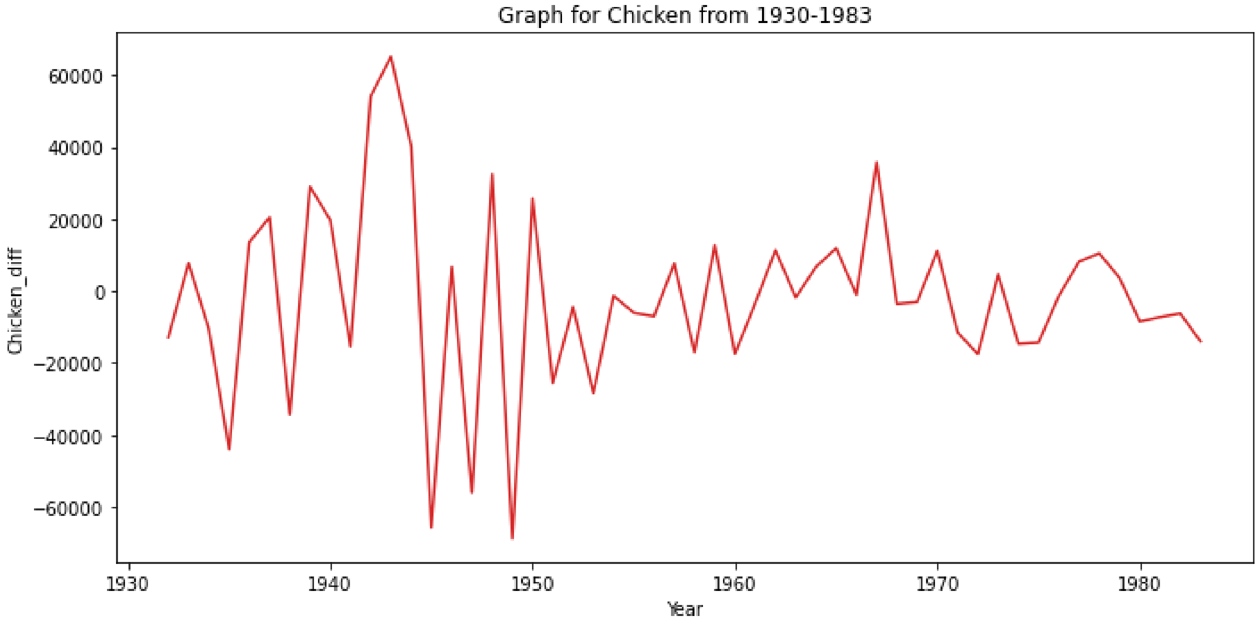
```python
result_diff1 = adfuller(df3['chicken_diff'])
print('ADF Statistic: %f' % result_diff1[0])
print('p-value: %f' %result_diff1[1])
print('Critical Values:')
for key, value in result_diff1[4].items():
    print('\t%s: %.3f' % (key, value))
print()
result_diff2 = adfuller(df3['egg in million_diff'])
print('ADF Statistic: %f' % result_diff2[0])
print('p-value: %f' % result_diff2[1])
print('Critical Values:')
for key, value in result_diff2[4].items():
    print('\t%s: %.3f' % (key, value))
print()
#Interpretation for first differencing for egg in million
if result_diff2[1] > 0.05:
    print("egg in million_diff is not stationary")
else:
    print("egg in million_diff is stationary")
#Interpretation for first differencing for chicken
if result_diff2[1] > 0.05:
    print("chicken_diff is not stationary")
else:
    print("chicken_diff is stationary")
```

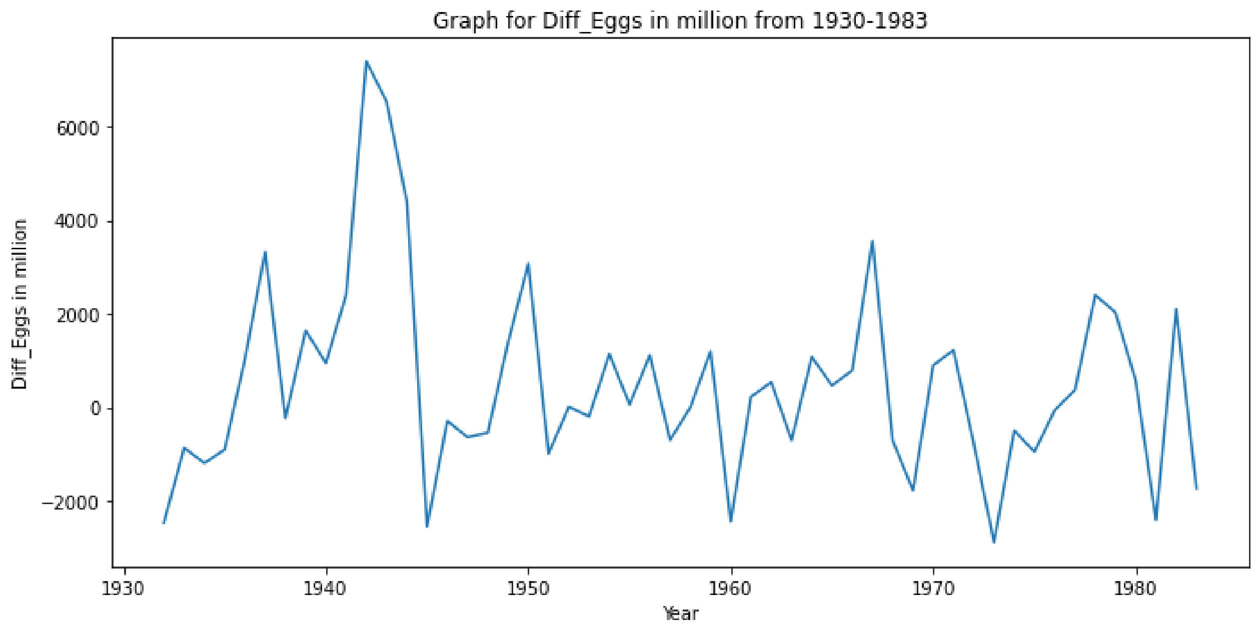
```
ADF Statistic: -3.744894
p-value: 0.003524
Critical Values:
	1%: -3.571
	5%: -2.923
	10%: -2.599

ADF Statistic: -4.874460
p-value: 0.000039
Critical Values:
	1%: -3.566
	5%: -2.920
	10%: -2.598

egg in million_diff is stationary
chicken_diff is stationary
```

In [252…

```python
fig, axs = plt.subplots(2, 1, figsize=(10, 10))
#graph for Differenced_Chickens from 1930-1983
axs[0].plot(df3['Year'], df3['chicken_diff'], 'tab:red')
axs[0].set_title('Graph for Chicken from 1930-1983')
axs[0].set_xlabel('Year')
axs[0].set_ylabel('Chicken_diff')
#graph for Differenced_Eggs from 1930-1983
axs[1].plot(df3['Year'], df3['egg in million_diff'], 'tab:blue')
axs[1].set_title('Graph for Diff_Eggs in million from 1930-1983')
axs[1].set_xlabel('Year')
axs[1].set_ylabel('Diff_Eggs in million')
plt.tight_layout()
plt.show()
```

```
df3.head(100)
df.reset_index(drop=True, inplace=True)
```





In [253…

```
#Since Egg in million and Chicken is stationary now then test for causality using grange

from statsmodels.tsa.stattools import grangercausalitytests
max_lags = 4
#Do Eggs granger cause chickens?
results = grangercausalitytests(df3[['chicken_diff', 'egg in million_diff']], max_lags,
```

```
Granger Causality
number of lags (no zero) 1
ssr based F test:         F=10.1404 , p=0.0025  , df_denom=48, df_num=1
ssr based chi2 test:   chi2=10.7742 , p=0.0010  , df=1
likelihood ratio test: chi2=9.7746  , p=0.0018  , df=1
parameter F test:         F=10.1404 , p=0.0025  , df_denom=48, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=4.3234  , p=0.0192  , df_denom=45, df_num=2
```

```
ssr based chi2 test:   chi2=9.6075  , p=0.0082  , df=2
likelihood ratio test: chi2=8.7879  , p=0.0124  , df=2
parameter F test:         F=4.3234  , p=0.0192  , df_denom=45, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=2.8636  , p=0.0480  , df_denom=42, df_num=3
ssr based chi2 test:   chi2=10.0228 , p=0.0184  , df=3
likelihood ratio test: chi2=9.1190  , p=0.0277  , df=3
parameter F test:         F=2.8636  , p=0.0480  , df_denom=42, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=4.0842  , p=0.0074  , df_denom=39, df_num=4
ssr based chi2 test:   chi2=20.1071 , p=0.0005  , df=4
likelihood ratio test: chi2=16.7942 , p=0.0021  , df=4
parameter F test:         F=4.0842  , p=0.0074  , df_denom=39, df_num=4
```

In [254…

```
lag_to_check = 4
significant_level = 0.05
p_value_lag_4 = results[lag_to_check][0]['ssr_ftest'][1]
```

In [257…

```
significant_level = 0.05
if p_value_lag_4 < significant_level:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is below the significa
else:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is above the significa
```

```
At lag 4, the p-value (0.00735712292576366) is below the significant level of 0.05, indi
cating statistical significance and that egg came first
```

In [256…

```
#The max lags to be used
max_lags = 4
#Do chicken granger cause eggs?
results = grangercausalitytests(df3[['egg in million_diff', 'chicken_diff']], max_lags,
```

```
Granger Causality
number of lags (no zero) 1
ssr based F test:         F=0.6786  , p=0.4141  , df_denom=48, df_num=1
ssr based chi2 test:   chi2=0.7210  , p=0.3958  , df=1
likelihood ratio test: chi2=0.7159  , p=0.3975  , df=1
parameter F test:         F=0.6786  , p=0.4141  , df_denom=48, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=0.3665  , p=0.6952  , df_denom=45, df_num=2
ssr based chi2 test:   chi2=0.8144  , p=0.6655  , df=2
likelihood ratio test: chi2=0.8078  , p=0.6677  , df=2
parameter F test:         F=0.3665  , p=0.6952  , df_denom=45, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=0.2080  , p=0.8903  , df_denom=42, df_num=3
ssr based chi2 test:   chi2=0.7279  , p=0.8666  , df=3
likelihood ratio test: chi2=0.7225  , p=0.8679  , df=3
parameter F test:         F=0.2080  , p=0.8903  , df_denom=42, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=0.2625  , p=0.9002  , df_denom=39, df_num=4
ssr based chi2 test:   chi2=1.2921  , p=0.8627  , df=4
```

```
likelihood ratio test: chi2=1.2751  , p=0.8656  , df=4
parameter F test:         F=0.2625  , p=0.9002  , df_denom=39, df_num=4
```

In [263…

```python
lag_to_check = 4
significant_level = 0.05
p_value_lag_4 = results[lag_to_check][0]['ssr_ftest'][1]
significant_level = 0.05
if p_value_lag_4 < significant_level:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is below the significa
else:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is above the significa
```

```
At lag 4, the p-value (0.9002291101518003) is above the significant level of 0.05, indic
ating no statistical significance that means chicken did not come first.
```

In [276…

```python
#To improve our research we would include data from 1930 to 2021
#Lets go back to the orginal data named df which is data from 1930-2021
#step 1: First differencing and second differencing if necessary.
#step 2: Granger Causality Test
df.head()
```

Out[276…

	Year	chicken	egg	egg in million
0	1930	468491	3581.0	42972.0
1	1931	449743	3532.0	42384.0
2	1932	436815	3327.0	39924.0
3	1933	444523	3255.0	39060.0
4	1934	433937	3156.0	37872.0

In [283…

```python
#Test for Stationarity using ADF test for egg in millions
result1 = adfuller(df['egg in million'])
result1 = adfuller(df['egg in million'].dropna())
print('ADF Statistic: %f' % result1[0])
print('p-value: %f' % result1[1])
print('Critical Values:')
for key, value in result1[4].items():
    print('\t%s: %.3f' % (key, value))
print()

#Test for Stationarity using ADF test for chicken
result2 = adfuller(df['chicken'].dropna())
result2 = adfuller(df['chicken'])
print('ADF Statistic: %f' % result2[0])
print('p-value: %f' % result2[1])
print('Critical Values:')
for key, value in result2[4].items():
    print('\t%s: %.3f' % (key, value))
print()

#Interpretation of egg in million result for the orignal data named df which is data fr
if result1[1] > 0.05:
    print("egg in million is not stationary")
else:
    print("egg in million is stationary")
```

```
#Interpretation of chicken result for the orignal data named df which is data from 1930

if result2[1] > 0.05:
    print("chicken is not stationary")
else:
    print("chicken is stationary")
```

```
ADF Statistic: 0.010383
p-value: 0.959382
Critical Values:
	1%: -3.505
	5%: -2.894
	10%: -2.584

ADF Statistic: -2.115193
p-value: 0.238449
Critical Values:
	1%: -3.508
	5%: -2.895
	10%: -2.585

egg in million is not stationary
chicken is not stationary
```

In [287…

```
# first order differencing for egg in million
df['egg in million_diff'] = df['egg in million'].diff().dropna()
# first order differencing for chicken
df['chicken_diff'] = df['chicken'].diff().dropna()

df['chicken_diff'] = df['chicken_diff'].dropna()

# Check and drop NaN values
df = df.dropna(subset=['chicken_diff'])
df = df.dropna(subset=['egg in million_diff'])
df.reset_index(drop=True, inplace=True)

aligned_years = df[df['Year'].isin(df.index)]
df.head()
```

Out[287…

	Year	chicken	egg	egg in million	egg in million_diff	chicken_diff
0	1934	433937	3156.0	37872.0	-1188.0	-10586.0
1	1935	389958	3081.0	36972.0	-900.0	-43979.0
2	1936	403446	3166.0	37992.0	1020.0	13488.0
3	1937	423921	3443.0	41316.0	3324.0	20475.0
4	1938	389624	3424.0	41088.0	-228.0	-34297.0

In [290…

```
#Test for Stationarity on the differenced data on egg in millions using ADF test
result_diff1 = adfuller(df['egg in million_diff'])
result_diff1 = adfuller(df['egg in million_diff'].dropna())
print('ADF Statistic: %f' % result_diff1[0])
print('p-value: %f' % result_diff1[1])
print('Critical Values:')
for key, value in result_diff1[4].items():
```

```python
    print('\t%s: %.3f' % (key, value))
print()

##Test for Stationarity on the differenced data on chicken using ADF test
result_diff2 = adfuller(df['chicken_diff'].dropna())
result_diff2 = adfuller(df['chicken_diff'])
print('ADF Statistic: %f' % result_diff2[0])
print('p-value: %f' % result_diff2[1])
print('Critical Values:')
for key, value in result_diff2[4].items():
    print('\t%s: %.3f' % (key, value))
print()

#Interpretation of egg in million result for the orignal data named df which is data fr
if result_diff1[1] > 0.05:
    print("egg in million_diff is not stationary")
else:
    print("egg in million_diff is stationary")

#Interpretation of chicken result for the orignal data named df which is data from 1930

if result_diff2[1] > 0.05:
    print("chicken_diff is not stationary")
else:
    print("chicken_diff is stationary")
```

```
ADF Statistic: -6.852031
p-value: 0.000000
Critical Values:
	1%: -3.508
	5%: -2.895
	10%: -2.585

ADF Statistic for chicken_diff: -2.778186
p-value for chicken_diff: 0.061458
Critical Values for chicken_diff:
	1%: -3.521
	5%: -2.901
	10%: -2.588

egg in million_diff is stationary
chicken_diff is not stationary
```

In [292...

```python
# Second order differencing for chicken
df['chicken_diff_2'] = df['chicken_diff'].diff().dropna()
df.head()
```

Out[292...

	Year	chicken	egg	egg in million	egg in million_diff	chicken_diff	chicken_diff_2
0	1934	433937	3156.0	37872.0	-1188.0	-10586.0	NaN
1	1935	389958	3081.0	36972.0	-900.0	-43979.0	-33393.0
2	1936	403446	3166.0	37992.0	1020.0	13488.0	57467.0
3	1937	423921	3443.0	41316.0	3324.0	20475.0	6987.0
4	1938	389624	3424.0	41088.0	-228.0	-34297.0	-54772.0

In [294…

```
#Lets drop the NaN values in the dataset
df = df.dropna(subset=['chicken_diff_2'])

df.reset_index(drop=True, inplace=True)

aligned_years = df[df['Year'].isin(df.index)]
df.head()
#Test for Stationarity on the Second differenced data on chicken using ADF test since i
result_diff2_2 = adfuller(df['chicken_diff_2'].dropna())
result_diff2_2 = adfuller(df['chicken_diff_2'])
print('ADF Statistic: %f' % result_diff2_2[0])
print('p-value: %f' % result_diff2_2[1])
print('Critical Values:')
for key, value in result_diff2_2[4].items():
    print('\t%s: %.3f' % (key, value))
print()

if result_diff2_2[1] > 0.05:
    print("chicken_diff_2 is not stationary")
else:
    print("chicken_diff_2 is stationary")
```

```
ADF Statistic: -4.589204
p-value: 0.000135
Critical Values:
	1%: -3.522
	5%: -2.901
	10%: -2.588

chicken_diff_2 is stationary
```

In [297…

```
#Since Eggs Chicken are Stationary at first differencing. Lets test for Causality using
max_lags = 4
#Do Eggs granger cause chickens?
Granger_results = grangercausalitytests(df[['chicken_diff_2', 'egg in million_diff']], 

lag_to_check = 4
significant_level = 0.05
p_value_lag_4 = Granger_results[lag_to_check][0]['ssr_ftest'][1]
```

```
Granger Causality
number of lags (no zero) 1
ssr based F test:         F=2.6170  , p=0.1095  , df_denom=83, df_num=1
ssr based chi2 test:   chi2=2.7116  , p=0.0996  , df=1
likelihood ratio test: chi2=2.6697  , p=0.1023  , df=1
parameter F test:         F=2.6170  , p=0.1095  , df_denom=83, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=4.4674  , p=0.0145  , df_denom=80, df_num=2
ssr based chi2 test:   chi2=9.4933  , p=0.0087  , df=2
likelihood ratio test: chi2=8.9996  , p=0.0111  , df=2
parameter F test:         F=4.4674  , p=0.0145  , df_denom=80, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=5.8733  , p=0.0012  , df_denom=77, df_num=3
ssr based chi2 test:   chi2=19.2218 , p=0.0002  , df=3
likelihood ratio test: chi2=17.3093 , p=0.0006  , df=3
parameter F test:         F=5.8733  , p=0.0012  , df_denom=77, df_num=3
```

```
Granger Causality
number of lags (no zero) 4
ssr based F test:         F=7.0223  , p=0.0001  , df_denom=74, df_num=4
ssr based chi2 test:   chi2=31.5056 , p=0.0000  , df=4
likelihood ratio test: chi2=26.7080 , p=0.0000  , df=4
parameter F test:         F=7.0223  , p=0.0001  , df_denom=74, df_num=4
```

In [298…

```
significant_level = 0.05
if p_value_lag_4 < significant_level:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is below the significa
else:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is above the significa
```

```
At lag 4, the p-value (7.546793785933029e-05) is below the significant level of 0.05, Do
not reject the null hypothesis and conclude that egg came first
```

In [299…

```
#Since Eggs Chicken are Stationary at first differencing. Lets test for Causality using
max_lags = 4
#Do chicken granger cause egg?
Granger_results = grangercausalitytests(df[['egg in million_diff', 'chicken_diff_2']], 

lag_to_check = 4
significant_level = 0.05
p_value_lag_4 = Granger_results[lag_to_check][0]['ssr_ftest'][1]
```

```
Granger Causality
number of lags (no zero) 1
ssr based F test:         F=0.0038  , p=0.9510  , df_denom=83, df_num=1
ssr based chi2 test:   chi2=0.0039  , p=0.9499  , df=1
likelihood ratio test: chi2=0.0039  , p=0.9499  , df=1
parameter F test:         F=0.0038  , p=0.9510  , df_denom=83, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=0.4884  , p=0.6154  , df_denom=80, df_num=2
ssr based chi2 test:   chi2=1.0379  , p=0.5951  , df=2
likelihood ratio test: chi2=1.0316  , p=0.5970  , df=2
parameter F test:         F=0.4884  , p=0.6154  , df_denom=80, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=0.2550  , p=0.8576  , df_denom=77, df_num=3
ssr based chi2 test:   chi2=0.8345  , p=0.8412  , df=3
likelihood ratio test: chi2=0.8304  , p=0.8422  , df=3
parameter F test:         F=0.2550  , p=0.8576  , df_denom=77, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=0.4682  , p=0.7589  , df_denom=74, df_num=4
ssr based chi2 test:   chi2=2.1006  , p=0.7173  , df=4
likelihood ratio test: chi2=2.0744  , p=0.7221  , df=4
parameter F test:         F=0.4682  , p=0.7589  , df_denom=74, df_num=4
```

In [300…

```
significant_level = 0.05
if p_value_lag_4 < significant_level:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is below the significa
else:
    print(f"At lag {lag_to_check}, the p-value ({p_value_lag_4}) is above the significa
```

At lag 4, the p-value (0.7588711669063641) is above the significant level of 0.05, reject the null hypothesis and conclude that means chicken did not come first.

In [ ]:

```
#In conclusion, expanding our data from "1930-1983" to "1930-1921", we found out that cl
#and conluded that eggs granger causes chicken. This conforms with the finding of Walter
```