1. Calculate mean, variance, standard deviation, and standard error of Close Price

```
import pandas as pd

infy_data = pd.read_csv('/content/infy data.csv')
infy_data
```

	Symbol	Date	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price
0	INFY	29-Jul-21	1605.60	1600.10	1620.00	1600.10	1615.50	1617.00
1	INFY	30-Jul-21	1617.00	1610.00	1623.40	1606.15	1611.75	1610.50
2	INFY	02-Aug-21	1610.50	1627.45	1634.75	1619.20	1632.00	1631.55
3	INFY	03-Aug-21	1631.55	1647.00	1658.65	1635.60	1657.00	1655.20
4	INFY	04-Aug-21	1655.20	1669.10	1679.90	1639.05	1648.55	1649.25
5	INFY	05-Aug-21	1649.25	1649.50	1666.00	1647.05	1652.00	1653.55
6	INFY	06-Aug-21	1653.55	1657.00	1660.65	1644.00	1652.10	1650.20
7	INFY	09-Aug-21	1650.20	1661.00	1667.45	1648.40	1664.40	1660.00

```
infy_data.describe()
```

	Prev Close	Open Price	High Price	Low Price	Last Price
count	11.000000	11.000000	11.000000	11.000000	11.000000
mean	1644.631818	1649.372727	1661.027273	1640.054545	1651.590909
std	25.224435	26.319207	25.226022	23.833731	23.771998
min	1605.600000	1600.100000	1620.000000	1600.100000	1611.750000
25%	1624.275000	1637.225000	1646.700000	1627.400000	1640.275000
50%	1650.200000	1657.000000	1666.000000	1644.000000	1652.100000
75%	1659.250000	1668.550000	1679.950000	1654.050000	1669.275000

Mean

```
#calculating mean without using built-in functions
no_of_items = len(infy_data['Close Price'])
mean = sum(infy_data['Close Price'])/no_of_items

print("Mean wihtout using built-in function is : ",mean)

#Using mean() built-in funciton
print("\nMean with built-in function is : ",infy_data['Close Price'].mean
```

```
Mean wihtout using built-in function is :  1652.2681818181816

Mean with built-in function is :  1652.2681818181816
```

Variance

```
#calculating deviations first then retriving variance through it (Without

deviations = []
for i in infy_data['Close Price']:
  deviations.append((i - mean) ** 2 )

variance = sum(deviations)/(no_of_items-1)
```

```
print("Variance without using built-in function is : ",variance)


#Using stat.variance() built-in function
import statistics as stat
print("\nVariance with built-in function is : ",stat.variance(infy_data['
```

```
Variance without using built-in function is :  621.9921363636355

Variance with built-in function is :  621.9921363636355
```

Standard Deviation

```
#calculating standard deviation without using built-in function
import math
standard_deviation = math.sqrt(variance)
print("Standard deviation without using built-in function is :", standard_


#using stat.stdev() built-in function
print("\nStandard deviation with built-in function is : ",stat.stdev(infy_
```

```
Standard deviation without using built-in function is : 24.939770174

Standard deviation with built-in function is :  24.93977017463544
```

Standard Error

```
#Calculating standard error without using built-in function
standard_error = standard_deviation/math.sqrt(no_of_items)
print("Standard error of without using built-in function is : ",standard_e

#using sem() built-in function
from scipy.stats import sem
print("\nStandard error with built-in function is : ",sem(infy_data['Close
```

```
Standard error of without using built-in function is :  7.519623638

Standard error with built-in function is :  7.51962363881474
```

2. Calculate Covariance and Correlation between Open Price and Close Price

Covariance

```
def covarfn(column1, column2, avg_column1, avg_column2):
    covariance = 0

    for i in range(0, len(column1)):
        covariance += (column1[i] - avg_column1) * (column2[i] - avg_colu
    return (covariance / (len(column1)-1))


avg_column1 = sum(infy_data["Open Price"])/len(infy_data["Open Price"])
avg_column2 = sum(infy_data["Close Price"])/len(infy_data["Close Price"])

#finding covariance using function
covariance_manual = covarfn(infy_data["Open Price"],infy_data["Close Price

#finding coariance using in-built function
covariance_inbuilt = infy_data["Open Price"].cov(infy_data["Close Price"]

print("Covairance between Open Price and Close Price without using built-i
print("\nCovariannce between Open Price and Close Price with built-in func
```

```
Covairance between Open Price and Close Price without using built-ir

Covariannce between Open Price and Close Price with built-in functic
```

Correlation

```
import math
def correlationCoefficient(col1, col2, n) :
    sum_col1 = 0
    sum_col2 = 0
    sum_both = 0
    squareSum_col1 = 0
```

```
    squareSum_col2 = 0
    i = 0
    while i < n :
        sum_col1 = sum_col1 + col1[i]
        sum_col2 = sum_col2 + col2[i]
        sum_both = sum_both + col1[i] * col2[i]

        squareSum_col1 = squareSum_col1 + col1[i] * col1[i]
        squareSum_col2 = squareSum_col2 + col2[i] * col2[i]
        i = i + 1

       #using formula for calculating correlation function
    corr = (float)(n * sum_both - sum_col1 * sum_col2)/(float)(math.sqrt(
    return corr


column_1 = infy_data["Close Price"]
column_2 = infy_data["Open Price"]

n = len(column1)
print("Correlation between Open Price and Close Price without using built

correlation = column_1.corr(column_2)
print("\nCorrelation between Open Price and Close Price with using built-
```

```
Correlation between Open Price and Close Price without using built-i

Correlation between Open Price and Close Price with using built-in f
```

3. Display graphically the distribution of samples.

```
import matplotlib.pyplot as plt
import seaborn as sb


plt.plot(infy_data["Date"],infy_data["Open Price"],label="Open Price");
plt.plot(infy_data["Date"],infy_data["Close Price"],label="Close Price");

plt.xticks(rotation = 90)


plt.title('Open Price vc Close Price')
```

```
plt.ylabel('Price ',fontsize=12);
plt.xlabel('Date',fontsize=13,labelpad=15);
plt.legend()
plt.show()
```