Nomor 2

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import shapiro
```

In [ ]:
```
df = pd.read_csv('banana.csv')
df = df.drop(df.columns[0], axis=1)
df_num = df.select_dtypes(include=['number'])
df_str = df.select_dtypes(include=['object'])
```

In [ ]:
```
IQR = {}
for column in df_num.columns:
    IQR[column] = np.percentile(df[column], 75) - np.percentile(df[column], 25)
```

In [ ]:
```
for column in df_num.columns:
  lowerBound = np.percentile(df_num[column], 25) - 1.5 * IQR[column]
  upperBound = np.percentile(df_num[column], 75) + 1.5 * IQR[column]
  print(f"{column}'s outlier with {lowerBound} lower bound and {upperBound} upperbo
  print(df_num[(df_num[column] < lowerBound) | (df_num[column] > upperBound)][colum
  print()
```

```
Acidity's outlier with 5.012314896354701 lower bound and 11.005988281432417 upperbou
nd (IQR: 1.4984183462694292)
148     11.191852
209     11.119288
279     11.137342
289     11.024219
345     11.079811
349     11.418636
683     11.026875
819      4.897068
966      4.456118
1040     4.896538
1327    11.284712
1785    11.374194
Name: Acidity, dtype: float64

Weight's outlier with 146.82637023654053 lower bound and 153.22835888037406 upperbou
nd (IQR: 1.6004971609583833)
44      146.535963
357     153.970493
386     146.376184
658     146.490788
677     146.444130
1059    154.070370
1116    146.603512
1133    146.496350
1159    146.126108
1269    153.285546
1412    146.812035
1793    146.060922
1898    146.533637
1959    153.599879
Name: Weight, dtype: float64

Length's outlier with 47.5082285751469 lower bound and 52.410305852223885 upperbound
(IQR: 1.225519319269246)
40      53.065151
446     52.413780
522     47.452026
637     52.543665
747     52.626968
792     47.313156
988     52.558423
1136    47.366597
1197    52.439588
1220    52.519990
1484    47.262146
1873    46.418052
Name: Length, dtype: float64

Appearance's outlier with 2.1647113424403055 lower bound and 7.747373338498701 upper
bound (IQR: 1.3956654990145987)
143     8.233968
242     2.127349
328     7.927957
594     7.842696
```

```
615     2.007510
1064    7.848426
1067    7.817189
1216    1.977268
1296    8.032614
1316    1.775864
1443    1.931581
1605    2.071613
1611    7.773449
1762    1.786403
1845    1.910726
Name: Appearance, dtype: float64

Tannin's outlier with 4.729826963771409 lower bound and 11.229598458769104 upperboun
d (IQR: 1.6249428737494238)
217     11.273264
400      4.291274
576      4.709272
581     12.090781
610      4.629238
687     11.780068
1261    11.431587
1456    12.416177
1461    11.550949
1484    11.250187
1631     4.650028
1796    11.355590
1989    11.521227
Name: Tannin, dtype: float64

Ripeness's outlier with 4.923425290238341 lower bound and 8.509645135586311 upperbou
nd (IQR: 0.8965549613369923)
233     8.767843
270     8.991369
280     8.645577
371     8.676075
427     8.628959
559     8.527220
757     8.637225
765     8.530369
822     9.482066
890     8.637212
901     8.629589
1028    4.862560
1121    9.173803
1142    8.636351
1288    8.698339
1300    9.348371
1353    8.573482
1373    9.114434
1493    8.834792
1507    4.904725
1567    8.612570
1633    8.707027
1675    4.918675
1693    8.782708
```

```
1881    9.425643
1956    8.539070
Name: Ripeness, dtype: float64

Sweetness's outlier with 4.448078990732531 lower bound and 8.074608633579198 upperbo
und (IQR: 0.9066324107116666)
29      4.025152
128     4.363350
143     4.053357
172     3.954111
186     4.411304
232     4.136793
329     4.151006
351     4.220835
418     4.179858
469     3.429437
791     4.339535
804     3.795591
1156    3.599487
1160    4.299325
1178    4.095918
1191    4.413483
1226    3.033193
1472    4.380152
1559    4.363427
1716    4.412548
1762    4.131909
Name: Sweetness, dtype: float64

Firmness's outlier with -0.5023027956582491 lower bound and 1.5154393695714752 upper
bound (IQR: 0.5044355413074311)
283    2.0
Name: Firmness, dtype: float64

Price's outlier with 19811.780900435893 lower bound and 20188.614577845517 upperboun
d (IQR: 94.20841935240605)
53      19803.813931
378     19781.569703
402         0.000000
689     19729.904103
690     19809.257798
759     20199.676334
789     20189.020997
832     19769.470304
873     19785.810537
964     19769.450553
995     19811.228690
1012    19759.846000
1095    19809.025516
1134    19763.590653
1294       -1.000000
1364    20281.431062
1474    19786.680740
1922        0.000000
Name: Price, dtype: float64
```

Pendefinisian outlier menurut kami adalah data yang lebih kecil dari `Q1 - 1.5 * IQR` atau lebih besar dari `Q3 + 1.5 * IQR`. Menurut kode diatas, dapat dilihat terdapat outlier pada semua atribut numerik. Tetapi sebelum ditindaklanjuti, harus dilihat terlebih dahulu apakah outlier yang ada merupakan data anomali atau memang benar data yang didapat di lapangan seperti itu.

Jika dilihat dan dibandingkan kembali dengan batas bawah dan batas atas, dapat dibilang bahwa kebanyakan data outlier bukanlah data anomali karena nilalinya yang tidak jauh berbeda dengan nilai batas bawah dan batas atas. Tetapi, pada atribut `Price` dan `Firmness`, terdapat data outlier yang anomali. Yaitu nilai `Price` -1 dan 0 dan nilai `Firmness` 2,0.

Selain itu, juga terdapat anomali pada atribut `Country_of_Origin`, yaitu data yang bernilai `undefined`.

Karena tidak ada outlier lain yang merupakan anomali, maka yang dihapus hanya empat nilai itu saja.

In [ ]:
```
df = df.drop(df[df['Price'] == 0].index)
df = df.drop(df[df['Price'] == -1].index)
df = df.drop(df[df['Firmness'] == 2.0].index)
df = df.drop(df[df['Country_of_Origin'] == 'undefined'].index)
df_num = df.select_dtypes(include=['number'])
df_str = df.select_dtypes(include=['object'])
```