In [1]:
```
from scipy.stats import median_abs_deviation
from scipy.stats import skew
from scipy.stats import norm, kurtosis
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
```

In [2]:
```
class_1 = [154,156,158,159,160,163,163,164,168,168,168,169,170,170,172,173,175,177,
```

In [74]:
```
Q1 = np.percentile(class_1,25,interpolation = 'midpoint')
Q3 = np.percentile(class_1,75,interpolation = 'midpoint')
QD_class1 = ((Q3-Q1)/2)
QD_class1
```

Out[74]:
```
5.5
```

In [75]:
```
Q1 = np.percentile(class_2,25,interpolation = 'midpoint')
Q3 = np.percentile(class_2,75,interpolation = 'midpoint')
QD_class2 = ((Q3-Q1)/2)
QD_class2
```

Out[75]:
```
2.0
```

In [77]:
```
print(f"QD calss 1: {QD_class1} and QD calss 2: {QD_class2}")
```

```
QD calss 1: 5.5 and QD calss 2: 2.0
```

In [4]:
```
mean_class = np.mean(class_1)
mean_class
```

Out[4]:
```
168.05
```

In [5]:
```
sk1 = skew(class_1)
sk1
```

Out[5]:
```
0.80762989890223
```

In [6]:
```
sd = np.std(class_1)
sd
```

Out[6]:
```
9.013739512544168
```

In [7]:
```
var = np.var(class_1)
var
```

Out[7]:
```
81.24749999999999
```

In [8]:
```
class_2 = [161,163,164,165,166,166,167,167,168,168,168,169,169,170,170,170,172,173,
In [9]:  sk2 = skew(class_2)
         sk2

Out[9]:  0.3244414017559051
```

```
In [10]: sd2 = np.std(class_2)
         sd2

Out[10]: 3.8131351929875232
```

```
In [11]: var2 = np.var(class_2)
         var2

Out[11]: 14.539999999999997
```

```
In [73]: print(f"skew class 1: {sk1:.1f} and skew class 2 : {sk2:.1f}")

         skew class 1: 0.8 and skew class 2 : 0.3
```

```
In [72]: print(f"std class 1: {sd:.1f} and std class 2 : {sd2:.1f}")

         std class 1: 9.0 and std class 2 : 3.8
```

```
In [71]: print(f"variance class 1: {var:.1f} and variance class 2 : {var2:.1f}")

         variance class 1: 81.2 and variance class 2 : 14.5
```

```
In [15]: med_class = np.median(class_1)
```

```
In [55]: med_class

Out[55]: 168.0
```

```
In [56]: med_class2 = np.median(class_2)
```

```
In [57]: med_class2

Out[57]: 168.0
```

In [54]:

```
def med(md,data):
    list_1=[]
    for i in data:
        jam = md-data
        jam = abs(jam)
        list_1.append(jam)
    a_2=0
    for l in list_1:
        jam_2 = l+a_2
        a_2=jam_2
    jam_ad_1 = jam_2//len(data)

    return jam_ad_1
b = med(med_class , class_1)
a_1=0
for i in b:
    jam_1=i+a_1
    a_1=jam_1
jam_ad = jam_1//len(class_1)
median_ad1 = jam_ad
print(median_ad1)
```

```
6.0
```

In [53]:

```
def med(md,data):
    list_1=[]
    for i in data:
        jam = md-data
        jam = abs(jam)
        list_1.append(jam)
    a_2=0
    for l in list_1:
        jam_2 = l+a_2
        a_2=jam_2
    jam_ad_1 = jam_2//len(data)

    return jam_ad_1
b = med(med_class , class_2)
a_1=0
for i in b:
    jam_1=i+a_1
    a_1=jam_1
jam_ad_1 = jam_1//len(class_1)
median_ad2 = jam_ad_1
print(median_ad2)
```

```
2.0
```

In [61]:

```
print(f"medianAD class 1: {median_ad1} and medianAD  class 2 : {median_ad2} with de
```

```
medianAD class 1: 6.0 and medianAD  class 2 : 2.0 with def
```

In [58]:

```
list_1=[]
for i in class_1:
    jam = med_class -i
    jam = abs(jam)
    list_1.append(jam)
    a=0
    for j in list_1:
        jam = j+a
        a=jam
medAD_class1 = jam//len(class_1)
print(medAD_class1)
```

```
6.0
```

In [59]:

```
list_2=[]
for i in class_2:
    jam =med_class2-i
    jam = abs(jam)
    list_2.append(jam)
    a=0
    for j in list_2:
        jam = j+a
        a=jam
medAD_class2 = jam//len(class_2)
print(medAD_class2)
```

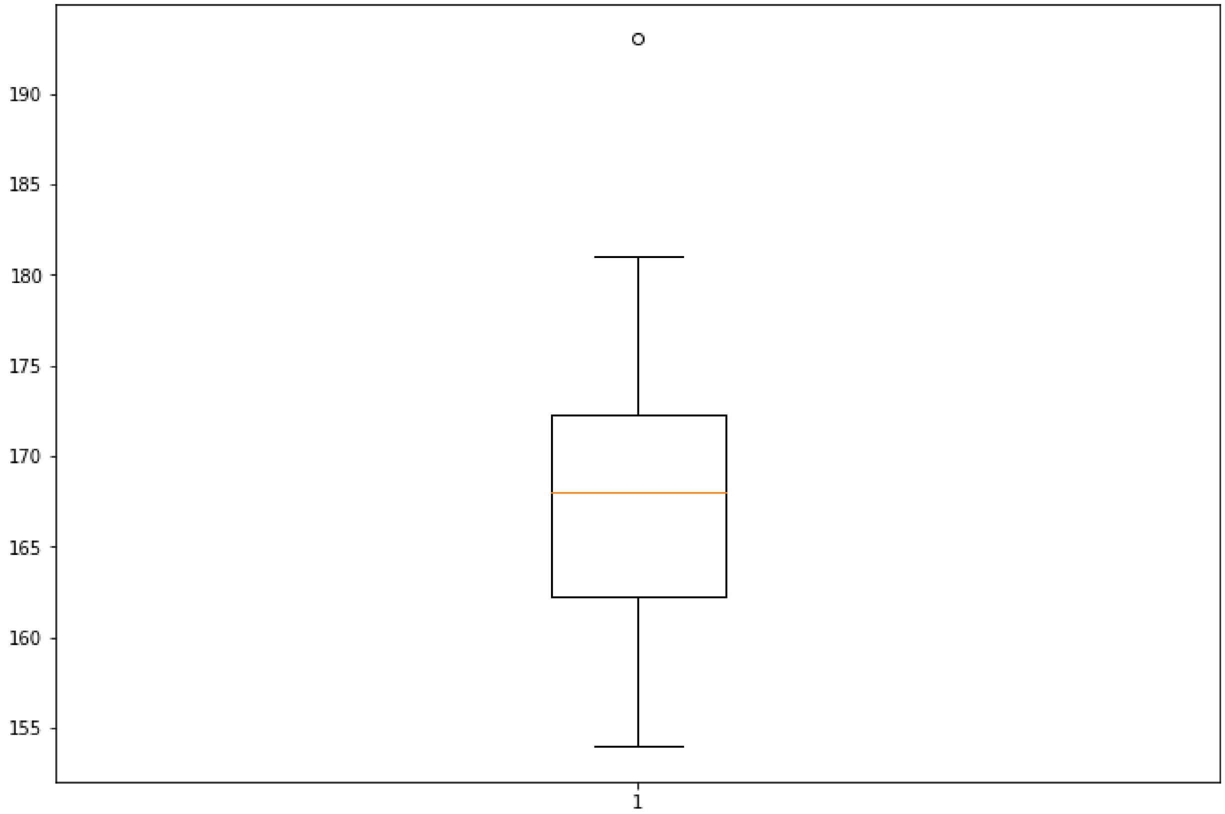
```
2.0
```

In [60]:

```
print(f"medianAD class 1: {medAD_class1} and medianAD  class 2 : {medAD_class2}")
```

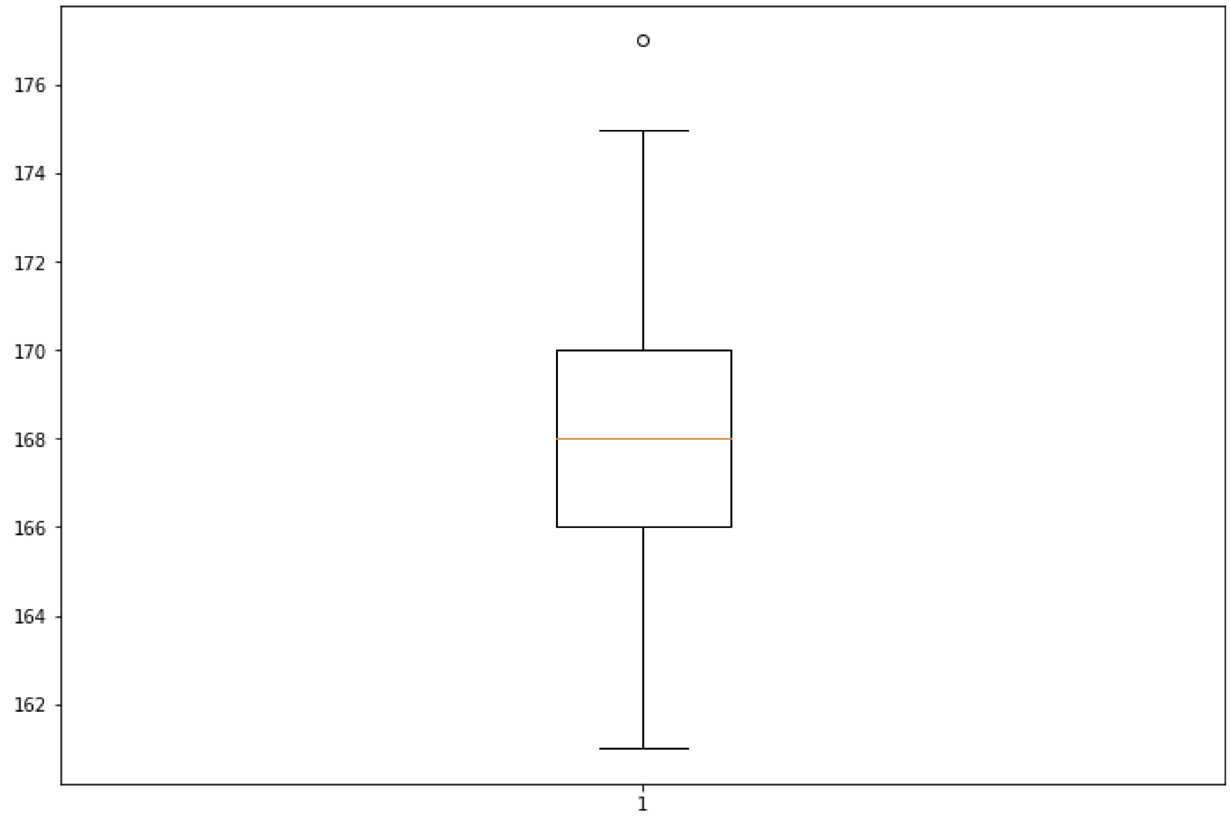
```
medianAD class 1: 6.0 and medianAD  class 2 : 2.0
```

In [62]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1.5,1.5])
box_1 = ax.boxplot(class_1)
plt.show()
```



In [66]:

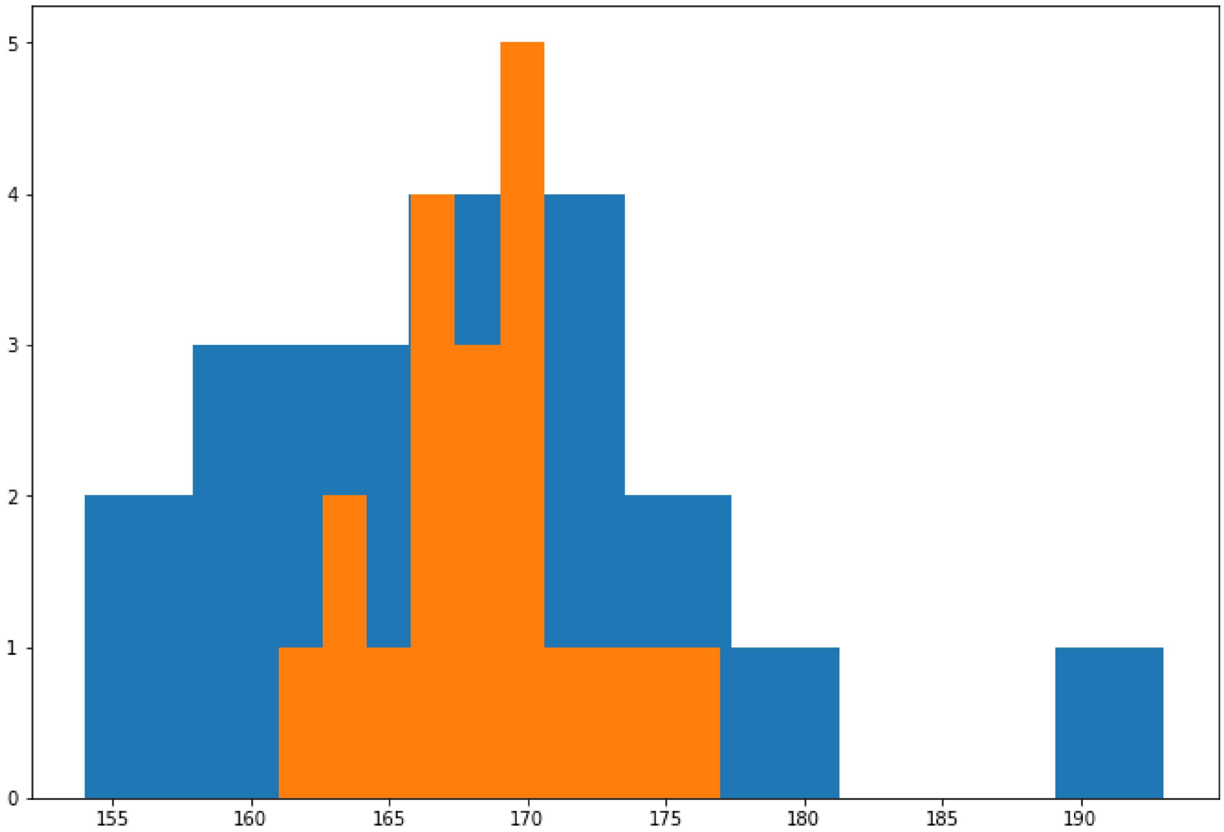
```
fig = plt.figure()
ax = fig.add_axes([0,0,1.5,1.5])
box_2 = ax.boxplot(class_2)
plt.show()
```



واریانس را نمیشود با نمودار تشخیص داد

```
In [64]: fig = plt.figure()
         ax = fig.add_axes([0,0,1.5,1.5])
         ax.hist(class_1, bins=10)
         ax.hist(class_2, bins=10)
```

```
Out[64]: (array([1., 2., 1., 4., 3., 5., 1., 1., 1., 1.]),
          array([161. , 162.6, 164.2, 165.8, 167.4, 169. , 170.6, 172.2, 173.8,
                 175.4, 177. ]),
          <BarContainer object of 10 artists>)
```



```
In [ ]:
```