Name : Ayush Makwana

Roll No : 18BCE107

Practical : 3

Implement suitable method (using concept of Quartile) in C/C++/Java/Python for detection of outliers present in the following data set : also take steps of remove these identified outliers from the given data set. Name Value

A-45 B-37 C-59 D-150 E-47 F-39 G-5 H-43 I-52 J-100

Numeric Outlier This is the simplest, nonparametric outlier detection method in a one dimensional feature space. Here outliers are calculated by means of the IQR (InterQuartile Range). The first and the third quartile (Q1, Q3) are calculated. An outlier is then a data point xi that lies outside the interquartile range. That is: Using the interquartile multiplier value k=1.5, the range limits are the typical upper and lower whiskers of a box plot.

xi > Q3 + k x IQR

xi < Q1 - k x IQR

In [18]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In [19]:
```
data = [45, 37, 59, 150, 47, 39, 5, 43, 52, 100]
sort_data = np.sort(data)
sort_data
```

Out[19]:
```
array([  5,  37,  39,  43,  45,  47,  52,  59, 100, 150])
```

In [20]:
```
Q1 = np.percentile(data, 25, interpolation = 'midpoint')
Q2 = np.percentile(data, 50, interpolation = 'midpoint')
Q3 = np.percentile(data, 75, interpolation = 'midpoint')

print('Q1 25 percentile of the given data is, ', Q1)
print('Q1 50 percentile of the given data is, ', Q2)
print('Q1 75 percentile of the given data is, ', Q3)

IQR = Q3 - Q1
print('Interquartile range is', IQR)
```

```
Q1 25 percentile of the given data is,  41.0
Q1 50 percentile of the given data is,  46.0
Q1 75 percentile of the given data is,  55.5
Interquartile range is 14.5
```

In [25]:
```
k=1.5
low_lim = Q1 - k * IQR
up_lim = Q3 + k * IQR
print('low_limit is', low_lim)
print('up_limit is', up_lim)
```

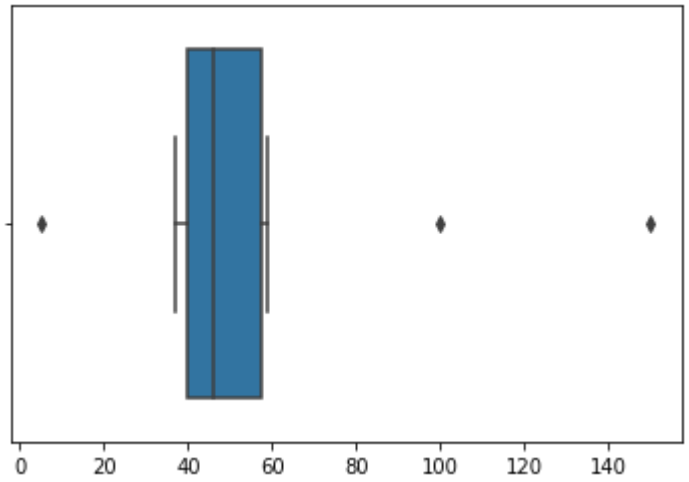
```
low_limit is 19.25
up_limit is 77.25
```

In [26]:
```
outlier =[]
for x in data:
    if ((x> up_lim) or (x<low_lim)):
         outlier.append(x)
print(' outlier in the dataset is', outlier)
```

```
 outlier in the dataset is [150, 5, 100]
```

In [27]:
```
sns.boxplot(data)
```

Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21e9234b088>
```



In [28]:
```
from scipy import stats
IQR = stats.iqr(data, interpolation = 'midpoint')
IQR
```

Out[28]:
```
14.5
```

Application

In [99]:
```
df = pd.read_csv('USA.csv')
df
```

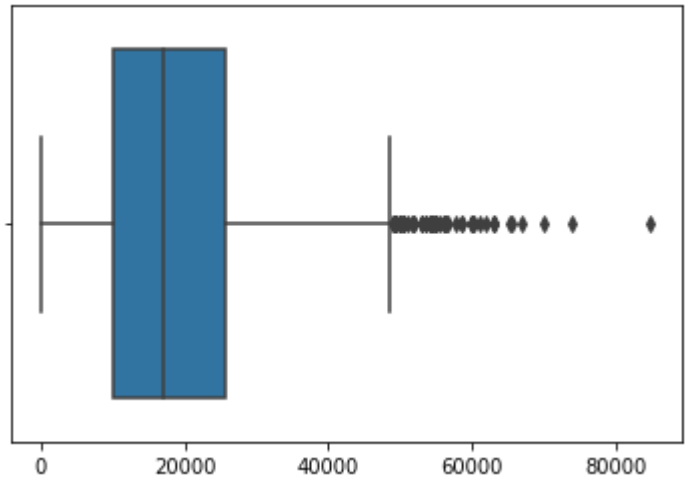
Out[99]:

	Unnamed: 0	price	brand
0	0	6300	toyota
1	1	2899	ford
2	2	5350	dodge
3	3	25000	ford
4	4	27700	chevrolet
...	...	...	...
2494	2494	7800	nissan
2495	2495	9200	nissan
2496	2496	9200	nissan
2497	2497	9200	nissan
2498	2498	9200	nissan

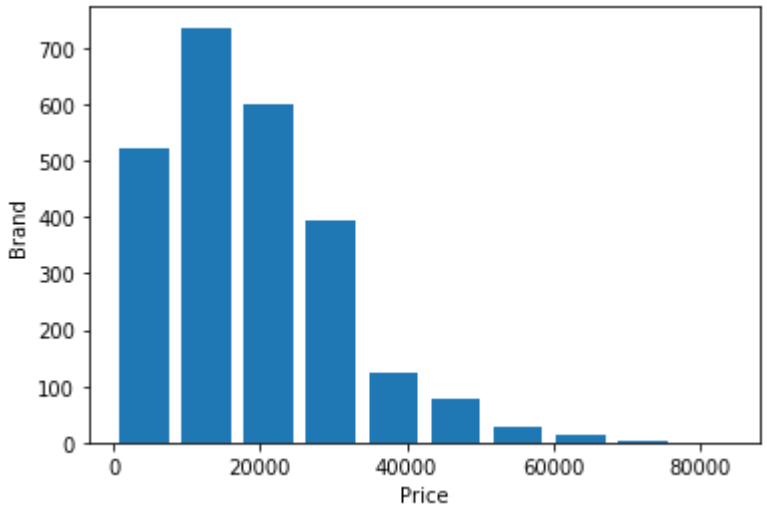
2499 rows × 3 columns

In [100]:
```
data1=df.to_numpy()
data1.shape
sns.boxplot(data1[:,1])
```

Out[100]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21e9265e2c8>
```



In [101]:
```
plt.hist(df.price, bins=10, rwidth=0.8)
plt.xlabel('Price')
plt.ylabel('Brand')
plt.show()
```



In [102]:
```
Q1 = df.price.quantile(0.25)
Q3 = df.price.quantile(0.75)
IQR = Q3 - Q1
print(Q1,Q3)
print(IQR)
```

```
10200.0 25555.5
15355.5
```

In [103]:
```
k=1.5
lwl = Q1 - k * IQR
upl = Q3 + k * IQR
print(lwl, upl)
```

```
-12833.25 48588.75
```

In [105]:
```
#outliers
outliers = df[(df.price < lwl) | (df.price > upl)]
print(outliers)
```

```
      Unnamed: 0  price      brand
44            44  55000       ford
49            49  54000       ford
95            95  53500        bmw
127          127  53000  chevrolet
277          277  67000      dodge
...          ...    ...        ...
2059        2059  49000       ford
2088        2088  59900       ford
2196        2196  50500       ford
2198        2198  55000       ford
2200        2200  56000       ford

[64 rows x 3 columns]
```

In [106]:
```
removed_outliers = df[(df.price > lwl) & (df.price < upl)]
removed_outliers
```

Out[106]:

	Unnamed: 0	price	brand
0	0	6300	toyota
1	1	2899	ford
2	2	5350	dodge
3	3	25000	ford
4	4	27700	chevrolet
...	...	...	...
2494	2494	7800	nissan
2495	2495	9200	nissan
2496	2496	9200	nissan
2497	2497	9200	nissan
2498	2498	9200	nissan

2435 rows × 3 columns

CONCLUSION

They give us an idea of what the data set looks like. Sometimes, looking at these values tell us whether the data is symmetrical or skewed.

In [ ]:
```

```