Statistics

```
In [2]: age = [23,24,32,12,43,67,45,32,56,32,160]
```

```
In [3]: import numpy as np

        np.mean(age)
```

```
Out[3]: 47.81818181818182
```

```
In [4]: np.median(age)
```

```
Out[4]: 32.0
```

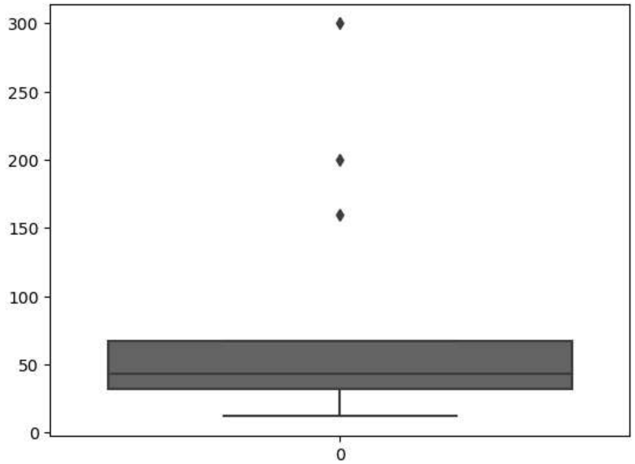
```
In [5]: import statistics as st
```

```
In [6]: stats = f'm : {st.mean(age):.3f} | md : {st.median(age)} | mo : {st.mode(age)}'

        stats
```

```
Out[6]: 'm : 47.818 | md : 32 | mo : 32'
```

```
In [7]: import seaborn as sns

        age = [23,24,32,12,43,67,45,32,56,32,160,200,300]
        sns.boxplot(age)
```

```
Out[7]: <Axes: >
```



5 no summary min, q1, median, q3, max

```
In [9]: q1, q3 = np.percentile(age,[25,75])
        iqr = q3 - q1
```

```
        lf = q1 - 1.5 * iqr
        hf = q3 + 1.5 * iqr
        md = np.median(age)

        print(f'lf: {lf} \nq1 :{q1}\nmd :{md}\nq3: {q3}\nhf: {hf}')
```

```
lf: -20.5
q1 :32.0
md :43.0
q3: 67.0
hf: 119.5
```

```
In [10]: def variance(data):
             n = len(data)
             mean = sum(data) / n
             deviation = [(i - mean)**2 for i in data]
             var = sum(deviation) / n
             return var
```

```
In [11]: variance(age)
```

```
Out[11]: 6957.30177514793
```

```
In [12]: np.var(age)
```

```
Out[12]: 6957.30177514793
```

```
In [13]: import math
         def standard_deviation(var):
             return math.sqrt(variance(age))
```

```
In [14]: standard_deviation(age)
```

```
Out[14]: 83.41044164340535
```

```
In [15]: np.std(age)
```

```
Out[15]: 83.41044164340535
```

Mean Square Error

```
In [17]: def mse(y_true, y_pred):
             try:
                 if len(y_true) == len(y_pred):
                     errors = [(y - h0)**2 for y, h0 in zip(y_true, y_pred)]
                     sum_of_square_error = sum(errors)
                     return sum_of_square_error / len(y_true)
                 else:
                     raise ValueError()
             except Exception as e:
                 print("Error! given y_true & y_pred are not same in size")
```

```
In [18]: y_actual = [5,4,1,2,10]
         y_predicted = [4,3.9,3,3.2,9.1]

         mse(y_actual,y_predicted)
```

```
Out[18]: 1.4520000000000002
```

Mean Absolute Error

```
In [20]: def mae(y_true, y_pred):
             try:
                 if len(y_true) == len(y_pred):
                     errors = [abs(y - h0) for y, h0 in zip(y_true, y_pred)]
                     sum_of_square_error = sum(errors)
                     return sum_of_square_error / len(y_true)
                 else:
                     raise ValueError()
             except Exception as e:
                 print("Error! given y_true & y_pred are not same in size")
```

```
In [21]: mae(y_actual,y_predicted)
```

```
Out[21]: 1.0400000000000003
```

Root Mean Square Error

```
In [23]: from  math import sqrt

         def rmse(y_true, y_pred):
             mse_value = mse(y_true, y_pred)
             return sqrt(mse_value)
```

```
In [24]: rmse(y_actual,y_predicted)
```

```
Out[24]: 1.2049896265113655
```

```
In [ ]:
```