26/9/25 - Python statistics Module

The **statistics** module in Python provides functions to perform **mathematical statistics** on numerical data.
It is widely used in **data analysis, mathematics, and machine learning**.

1. Mean (Average)

```
import statistics

data = [10, 20, 30, 40, 50]
print("Mean:", statistics.mean(data))
```

Output:

```
Mean: 30
```

🔎 **Theory**

- **Mean = (Sum of all values) ÷ (Number of values)**
- Example: (10 + 20 + 30 + 40 + 50) ÷ 5 = 30.

2. Median (Middle Value)

```
import statistics

data = [5, 7, 9, 11, 13]
print("Median:", statistics.median(data))
```

Output:

```
Median: 9
```

🔎 **Theory**

- The **median** is the middle value when data is sorted.
- If data has even numbers, median = average of two middle values.

3. Mode (Most Frequent Value)

```
import statistics

data = [1, 2, 2, 3, 4, 4, 4, 5]
print("Mode:", statistics.mode(data))
```

Output:

```
Mode: 4
```

🔎 **Theory**

- **Mode** is the value that occurs most frequently.
- Example: In [1,2,2,3,4,4,4,5], mode = 4 because it appears 3 times.

4. Standard Deviation (Spread of Data)

```
import statistics

data = [10, 20, 30, 40, 50]
print("Standard Deviation:", statistics.stdev(data))
```

Output:

```
Standard Deviation: 15.811388300841896
```

🔎 **Theory**

- **Standard Deviation (stdev)** measures how much values deviate from the mean.
- A **low stdev** → values are close to mean.
- A **high stdev** → values are spread out.

5. Variance (Square of Standard Deviation)

```
import statistics

data = [10, 20, 30, 40, 50]
print("Variance:", statistics.variance(data))
```

Output:

Variance: 250

🔎 **Theory**

- **Variance** = Average of squared differences from the mean.
- It shows the **spread of data**, like stdev but in squared units.

6. Harmonic Mean

```
import statistics

data = [2, 4, 4]
print("Harmonic Mean:", statistics.harmonic_mean(data))
```

Output:

Harmonic Mean: 3.0

🔎 **Theory**

- **Harmonic Mean** = $n \div (1/x_1 + 1/x_2 + \ldots + 1/x_n)$.
- Used when working with **rates, speeds, and ratios**.

7. Median Low & Median High

```
import statistics

data = [10, 20, 30, 40]
print("Median Low:", statistics.median_low(data))
print("Median High:", statistics.median_high(data))
```

Output:

Median Low: 20
Median High: 30

🔎 **Theory**

- If dataset has even number of values:
    - **median_low()** → Returns lower middle value.
    - **median_high()** → Returns higher middle value.

8. Quantiles

```
import statistics

data = [10, 20, 30, 40, 50, 60, 70, 80]
print("Quantiles:", statistics.quantiles(data, n=4))
```

Output:

```
Quantiles: [27.5, 45.0, 62.5]
```

🔎 **Theory**

- **Quantiles** divide data into equal parts.
- With n=4, it divides data into **quartiles**.
- Example: [27.5, 45.0, 62.5] means Q1 = 27.5, Q2 = 45, Q3 = 62.5.