Statistics Basics

Assignment Q/A

# 1. What is statistics, and why is it important?

Statistics is the science of collecting, organizing, analyzing, interpreting, and presenting data to make informed decisions or understand patterns.

Why It Is Important:

- Informed Decisions – Helps in making choices based on data, not guesswork.

- Pattern Recognition – Identifies trends and relationships in data.

- Research Support – Essential in experiments and scientific studies.

- Risk Reduction – Assists in forecasting and managing uncertainty.

- RealWorld Use – Widely used in business, healthcare, government, education, etc.

In short, statistics turns raw data into useful information for better understanding and decision-making.

# 2. What are the two main types of statistics ?

The two main types of statistics are:

- Descriptive Statistics – Summarizes and presents data using charts, averages, etc.

- Inferential Statistics – Makes predictions or conclusions about a population based on a sample.

# 3. What are descriptive statistics ?

Descriptive statistics are statistical methods used to summarize, organize, and describe the main features of a dataset. They help present data in a meaningful way using:

- Measures of central tendency: Mean, median, mode

- Measures of dispersion: Range, variance, standard deviation

- Data presentation tools: Tables, charts, graphs

These statistics do not draw conclusions beyond the data but provide a clear overview of it.

# 4. What is inferential statistics?

Inferential statistics is the branch of statistics that involves using data from a sample to make predictions, estimates, or decisions about a larger population.

It includes techniques like:

- Hypothesis testing

- Confidence intervals

- Regression analysis

- Chi-square tests, etc.

In short, inferential statistics helps us draw conclusions and make generalizations beyond the actual data collected.

# 5. What is sampling in statistics?

Sampling in statistics is the process of selecting a small group (sample) from a larger population to study and analyze. This allows researchers to draw conclusions about the whole population without examining every individual.

Why Sampling is Important:

- Saves time and cost

- Makes large-scale studies more practical

- Still provides reliable insights if done properly

Example:

Surveying 1,000 voters to predict the outcome of a national election instead of asking every voter.

# 6. What are the different types of sampling methods ?

There are two main categories of sampling methods: probability sampling and non-probability sampling.

1. Probability Sampling:

Each member of the population has a known, non-zero chance of being selected.

- Simple Random Sampling: Every individual has an equal chance of being chosen.

- Systematic Sampling: Selecting every kth individual from a list (e.g., every 10th person).

- Stratified Sampling: Population is divided into groups (strata), and samples are taken from each group.

- Cluster Sampling: Population is divided into clusters (usually by location), and whole clusters are randomly selected.

2. Non-Probability Sampling:

Not every individual has a known or equal chance of being selected.

- Convenience Sampling: Selecting individuals who are easiest to reach.

- Judgmental/Purposive Sampling: Selecting based on the researcher's judgment.

- Quota Sampling: Ensuring specific characteristics are represented in set proportions.

- Snowball Sampling: Existing subjects recruit future subjects, often used for hard-to-reach populations.

Each method has its own advantages and is chosen based on the research goal, population size, and resources available.

# 7. What is the difference between random and non-random sampling ?

Random sampling = more accurate, unbiased results

Non-random sampling = easier, quicker, but less reliable

# 8. Define and give examples of qualitative and quantitative data ?

Qualitative Data

Definition: Non-numerical data that describes qualities or characteristics.

Examples:

- Colors (red, blue)
- Types of fruits (apple, banana)
- Opinions (happy, sad)
- Gender (male, female)

Quantitative Data

Definition: Numerical data that can be measured or counted.

Examples:

- Age (25, 30, 40)
- Height (150 cm, 180 cm)
- Number of students (50, 100)
- Temperature (20°C, 30°C)

Summary:

Qualitative = descriptive, categorical

Quantitative = numerical, measurable

# 9. What are the different types of data in statistics ?

1. Quantitative (Numerical) Data
Data that represents measurable quantities.

Discrete: Countable values, usually whole numbers
Example: Number of students, number of cars

Continuous: Any value within a range, including decimals
Example: Height, weight, temperature

2. Qualitative (Categorical) Data
Data that describes categories or qualities, not numbers.

Nominal: Categories without any order
Example: Colors (red, blue), Gender (male, female)

Ordinal: Categories with a meaningful order but no fixed interval
Example: Ratings (poor, fair, good), Education level (high school, bachelor, master)

# 10. Explain nominal, ordinal, interval, and ratio levels of measurement ?

1. Nominal Level

Definition: Data is categorized without any order or ranking.

Example: Gender (male, female), Blood type (A, B, AB, O), Colors (red, blue).

Key Point: Categories are labels only; no mathematical operations apply.

2. Ordinal Level

Definition: Data is categorized with a meaningful order, but differences between categories are not necessarily equal.

Example: Movie ratings (poor, fair, good), Education level (high school, bachelor, master).

Key Point: You know the order but not the exact difference between ranks.

3. Interval Level

Definition: Data has ordered categories with equal intervals between values, but no true zero point.

Example: Temperature in Celsius or Fahrenheit (difference matters, but zero doesn’t mean “none”).

Key Point: You can add and subtract, but ratios don't make sense (e.g., 20°C is not "twice as hot" as 10°C).

4. Ratio Level

Definition: Like interval data, but with a true zero point representing absence of the quantity.

Example: Height, weight, age, income.

Key Point: You can do all arithmetic operations, including meaningful ratios (e.g., 40 kg is twice 20 kg).

# 11. What is the measure of central tendency ?

Measure of central tendency refers to statistical values that represent the center or typical value of a dataset. They summarize a large set of data by identifying a single value that best describes the entire distribution.

The three main measures are:

Mean – The average of all data points (sum of values ÷ number of values).

Median – The middle value when data is arranged in order.

Mode – The value that appears most frequently in the dataset.

# 12. Define mean, median, and mode.

Mean: The average of all numbers in a dataset. Calculated by adding all values and dividing by the total number of values.
Example: For [2, 4, 6], mean = (2+4+6)/3 = 4.

Median: The middle value when the data is arranged in ascending or descending order. If there's an even number of values, it's the average of the two middle values.
Example: For [2, 4, 6], median = 4; for [2, 4, 6, 8], median = (4+6)/2 = 5.

Mode: The value that occurs most frequently in the dataset. There can be more than one mode or none if all values are unique.
Example: For [2, 4, 4, 6], mode = 4.

# 13. What is the significance of the measure of central tendency ?

The measure of central tendency is important because it summarizes data with a single typical value, making it easier to understand, compare, and make decisions based on the data.

# 14. What is variance, and how is it calculated ?

Variance is a measure of how much the data values vary or spread out from the mean.

How to calculate variance:

1. Find the mean (average) of the data.

2. Subtract the mean from each data point and square the result.

3. Find the average of these squared differences.

Formula for population variance: and **Formula for sample variance:**

$$\sigma^2 = \frac{\Sigma(x-\mu)^2}{N} \quad \text{Population Variance}$$

$$s^2 = \frac{\Sigma(x-\bar{x})^2}{n-1} \quad \text{Sample Variance}$$

Population	Sample
$\sigma^2 = \frac{\Sigma(x_i-\mu)^2}{n}$	$S^2 = \frac{\Sigma(x_i-\overline{x})^2}{n-1}$
μ - Population Average	x - Sample Average
xi - Individual Population Value	x_i - Individual Population Value
n - Total Number of Population	n - Total Number of Sample
σ^2 - Variance of Population	S^2 - Variance of Sample

# 15. What is standard deviation, and why is it important ?

Standard deviation is a measure that shows how much the data values vary or spread out from the mean. It is the square root of the variance.

Why is it important?

- It tells us how spread out or clustered the data is around the average.

- A small standard deviation means data points are close to the mean.

- A large standard deviation means data points are more spread out.

- It is used widely in statistics, finance, science, and many other fields to understand variability and risk.

# 16. Define and explain the term range in statistics.

Range in statistics is the simplest measure of data spread. It is the difference between the largest and smallest values in a dataset.

How to calculate range:

Range = Maximum value − Minimum value

Example:

If the data set is [5, 8, 12, 20, 3],

- Maximum value = 20

- Minimum value = 3

- Range = 20 − 3 = 17

Significance:

- Range shows the overall spread or dispersion of data.

- It is easy to calculate but can be affected by extreme values (outliers).

# 17. What is the difference between variance and standard deviation?

Variance

- .Variance measures how much the data points spread out from the mean.

- It is calculated as the average of the squared differences from the mean.

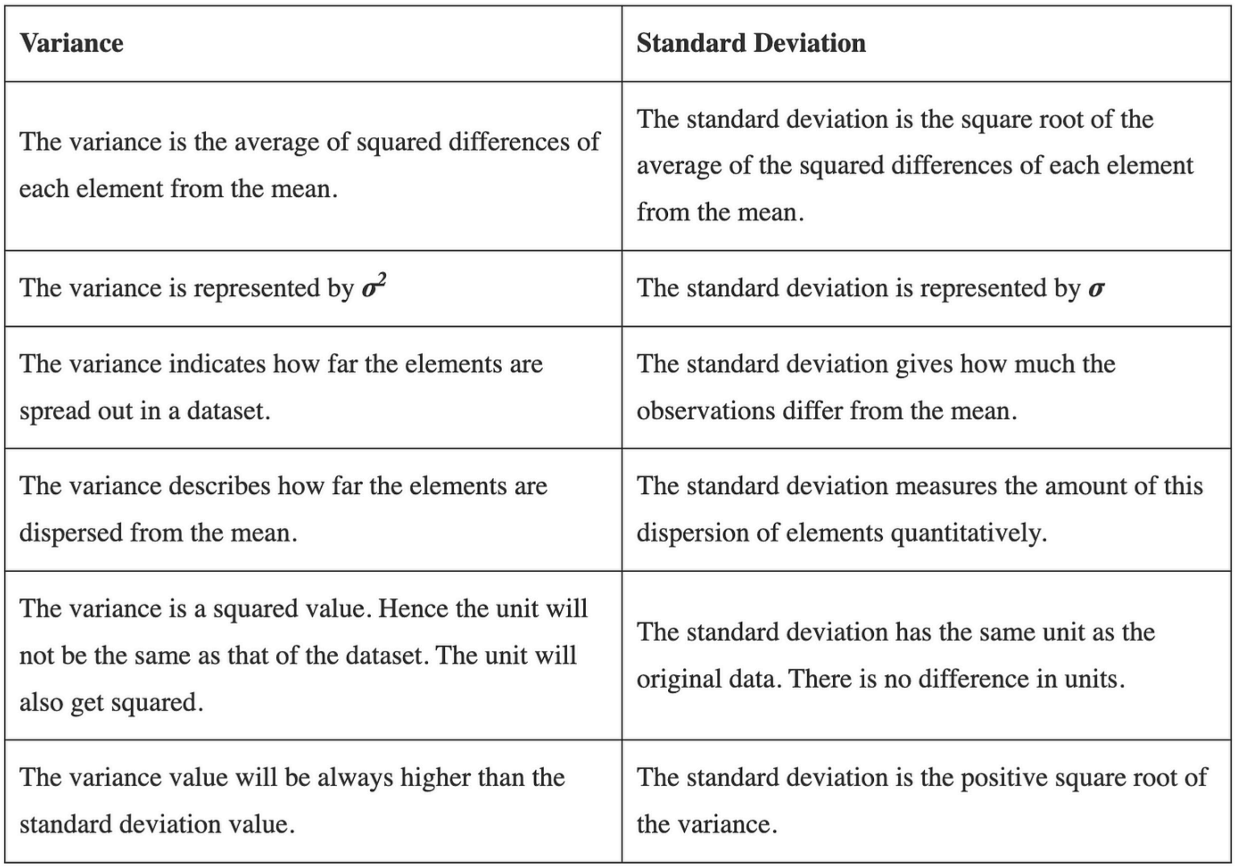
- Variance is expressed in squared units of the original data (e.g., if data is in meters, variance is in meters^2).

Standard Deviation

- Standard deviation is the square root of the variance.

- It shows the spread of data points in the same units as the original data.

- It is easier to interpret because it relates directly to the data values.



Variance	**Standard Deviation**
The variance is the average of squared differences of each element from the mean.	The standard deviation is the square root of the average of the squared differences of each element from the mean.
The variance is represented by $\boldsymbol{\sigma}^2$	The standard deviation is represented by $\boldsymbol{\sigma}$
The variance indicates how far the elements are spread out in a dataset.	The standard deviation gives how much the observations differ from the mean.
The variance describes how far the elements are dispersed from the mean.	The standard deviation measures the amount of this dispersion of elements quantitatively.
The variance is a squared value. Hence the unit will not be the same as that of the dataset. The unit will also get squared.	The standard deviation has the same unit as the original data. There is no difference in units.
The variance value will be always higher than the standard deviation value.	The standard deviation is the positive square root of the variance.

# 18. What is skewness in a dataset ?

Skewness in a dataset measures the asymmetry or lack of symmetry in the distribution of data values.

Explanation:

- If the data is symmetrical, skewness is zero (like a perfect bell curve).

- If the data has a long tail on the right (more values on the left), it is positively skewed (right-skewed).

- If the data has a long tail on the left (more values on the right), it is negatively skewed (left-skewed).

In short: Skewness shows whether data leans more to one side or is balanced.

#19. What does it mean if a dataset is positively or negatively skewed ?

1.Positively Skewed (Right-Skewed):

- The tail is longer on the right side.

- Most values are concentrated on the left, with a few large values pulling the mean to the right.

- Mean > Median > Mode

Example: Income data — a few people earn very high incomes.

2.Negatively Skewed (Left-Skewed):

- The tail is longer on the left side.

- Most values are concentrated on the right, with a few small values pulling the mean to the left.

- Mean < Median < Mode

Example: Test scores where most students score high, but a few score very low.

In short:

- Positive skew → tail on right, mean > median

- Negative skew → tail on left, mean < median

# 20. Define and explain kurtosis.

Kurtosis is a statistical measure that describes the shape of a distribution's tails in relation to its overall shape, especially how heavy or light the tails are compared to a normal distribution.

Types of Kurtosis:

1.Mesokurtic:

- Normal distribution

- Tails are moderate (not too heavy or light)

- Kurtosis ≈ 3 (Excess kurtosis = 0)

2.Leptokurtic:

- Peaked with heavy tails

- More data in the tails and center

- Kurtosis > 3 (Excess kurtosis > 0)

Example: Stock market returns

3.Platykurtic:

- Flat distribution with light tails

- Less data in the tails

- Kurtosis < 3 (Excess kurtosis < 0)

Example: Uniform distribution

# 21. What is the purpose of covariance ?

Purpose of Covariance:

Covariance measures the direction of the relationship between two variables.

### What it tells us:

- Whether two variables increase or decrease together.

- The strength and direction of their relationship.

Interpretation:

1. Positive covariance:

When one variable increases, the other tends to increase.
(e.g., height and weight)

2. Negative covariance:

When one variable increases, the other tends to decrease.
(e.g., hours of study and number of errors)

3. Zero covariance:

No consistent pattern or relationship.

# 22. What does correlation measure in statistics ?

Correlation in statistics measures the strength and direction of the linear relationship between two variables.

# What it tells us:

1. Positive correlation:

As one variable increases, the other also increases.
(e.g., height and weight)

2. Negative correlation:

As one variable increases, the other decreases.
(e.g., speed and travel time)

3. No correlation:

No relationship between the variables.

# Value Range:

Correlation coefficient (r) ranges from –1 to +1

+1 = perfect positive correlation

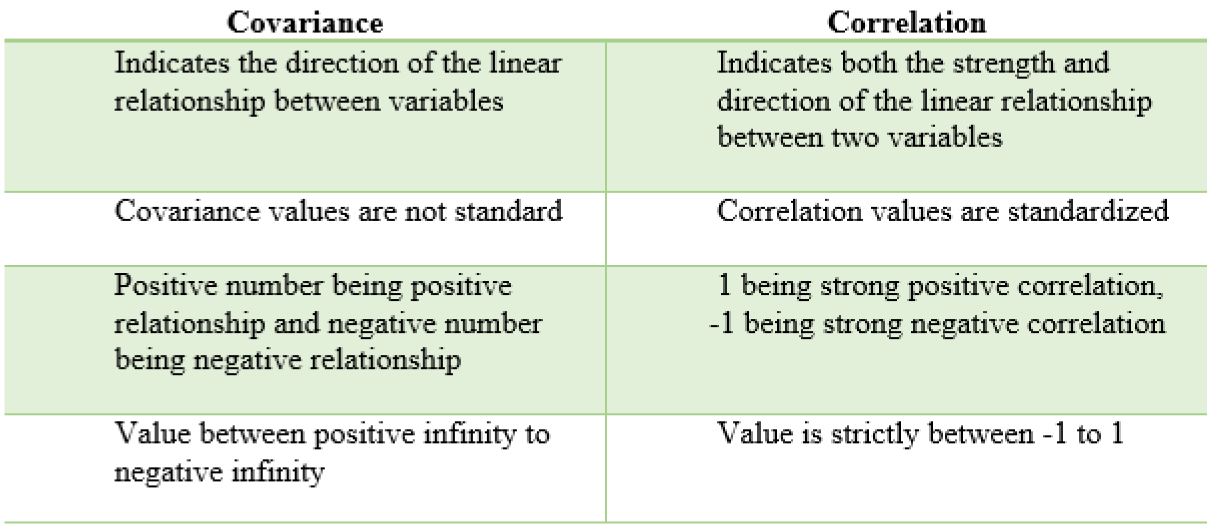
–1 = perfect negative correlation

0 = no correlation

# In short:

Correlation shows how closely two variables move together in a straight-line relationship.

# 23. What is the difference between covariance and correlation ?



Covariance	Correlation
Indicates the direction of the linear relationship between variables	Indicates both the strength and direction of the linear relationship between two variables
Covariance values are not standard	Correlation values are standardized
Positive number being positive relationship and negative number being negative relationship	1 being strong positive correlation, -1 being strong negative correlation
Value between positive infinity to negative infinity	Value is strictly between -1 to 1

24. Here are some important real-world applications of statistics:

Here are some important real-world applications of statistics:

📊 1. Business & Marketing:

- Analyzing customer data to improve sales.

- Forecasting demand and setting prices.

- A/B testing for product or website changes.

🏥 2. Healthcare:

- Analyzing clinical trial results.

- Tracking disease outbreaks.

- Measuring the effectiveness of treatments.

📚 3. Education:

- Evaluating student performance and school rankings.

- Designing standardized tests.

- Understanding learning trends.

🧠 4. Psychology & Social Sciences:

- Conducting surveys and experiments.

- Analyzing human behavior patterns.

📈 5. Finance & Economics:

- Stock market analysis.

- Risk assessment and portfolio management.

- Economic forecasting (like inflation or unemployment rates).

🌍 6. Government & Public Policy:

- Census data analysis.

- Policy planning and resource allocation.

- Monitoring crime, poverty, and health statistics.

🏗️ 7. Manufacturing & Quality Control:

- Ensuring product quality using control charts.

- Reducing defects and improving processes.

Practical

```
# 1. How do you calculate the mean, median, and mode of a dataset ?
import statistics

data = [4, 8, 6, 5, 3, 9, 8, 8]

mean = statistics.mean(data)
median = statistics.median(data)
mode = statistics.mode(data)

print("Mean:", mean)
print("Median:", median)
print("Mode:", mode)
```

```
Mean: 6.375
Median: 7.0
Mode: 8
```

```
# 2. Write a Python program to compute the variance and standard deviation of a dataset.
```

```
import statistics

# Sample dataset
data = [10, 12, 23, 23, 16, 23, 21, 16]

variance = statistics.variance(data)  # Sample variance
std_dev = statistics.stdev(data)  # Sample standard deviation

print("Variance:", variance)
print("Standard Deviation:", std_dev)
```

```
Variance: 27.428571428571427
Standard Deviation: 5.237229365663818
```

```
import numpy as np

# Sample dataset
data = np.array([10, 12, 23, 23, 16, 23, 21, 16])

variance = np.var(data, ddof=1)  # ddof=1 for sample variance
std_dev = np.std(data, ddof=1)  # ddof=1 for sample std deviation

print("Variance:", variance)
print("Standard Deviation:", std_dev)
```

```
Variance: 27.428571428571427
Standard Deviation: 5.237229365663817
```

# 3. Create a dataset and classify it into nominal, ordinal, interval, and ratio types.

```
import pandas as pd

# Creating a sample dataset
data = {
   "Name": ["Alice", "Bob", "Charlie", "Diana"],
   "Education_Level": ["High School", "Bachelor", "Master", "PhD"],  # Ordinal
   "Temperature_C": [22, 25, 24, 23],  # Interval
   "Salary": [40000, 52000, 61000, 72000],  # Ratio
   "Department": ["HR", "Finance", "IT", "Marketing"]  # Nominal
}

df = pd.DataFrame(data)
```

```
# Display the dataset
print("Dataset:\n")
print(df)

# Classification explanation
print("\nData Type Classification:")
print("- Name: Nominal (Labels without order)")
print("- Education_Level: Ordinal (Has order: High School < Bachelor < Master < PhD)")
print("- Temperature_C: Interval (Has order & difference, but no true zero)")
print("- Salary: Ratio (Has order, true zero, and ratios make sense)")
print("- Department: Nominal (Categories without order)")
```

```
Dataset:

      Name Education_Level  Temperature_C  Salary Department
0    Alice     High School             22   40000         HR
1      Bob        Bachelor             25   52000    Finance
2  Charlie          Master             24   61000         IT
3    Diana             PhD             23   72000  Marketing

Data Type Classification:
- Name: Nominal (Labels without order)
- Education_Level: Ordinal (Has order: High School < Bachelor < Master < PhD)
- Temperature_C: Interval (Has order & difference, but no true zero)
- Salary: Ratio (Has order, true zero, and ratios make sense)
- Department: Nominal (Categories without order)
```

# 4. Implement sampling techniques like random sampling and stratified sampling

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Sample dataset
data = {
  'ID': range(1, 21),
  'Age': [22, 25, 47, 35, 46, 52, 23, 43, 36, 29, 40, 33, 26, 31, 38, 45, 28, 50, 41, 39],
  'Gender': ['M', 'F', 'M', 'F', 'F', 'M', 'M', 'F', 'M', 'F', 'F', 'M', 'F', 'M', 'F', 'M', 'F', 'M', 'F', 'M'],
  'Department': ['HR', 'Finance', 'IT', 'IT', 'Finance', 'HR', 'IT', 'Finance', 'HR', 'Finance',
                 'HR', 'IT', 'Finance', 'IT', 'HR', 'Finance', 'IT', 'HR', 'Finance', 'IT']
}

df = pd.DataFrame(data)
print("Original Dataset:\n", df)
```

```
Original Dataset:
     ID  Age Gender Department
0    1   22      M         HR
1    2   25      F    Finance
2    3   47      M         IT
3    4   35      F         IT
4    5   46      F    Finance
5    6   52      M         HR
6    7   23      M         IT
7    8   43      F    Finance
8    9   36      M         HR
9   10   29      F    Finance
10  11   40      F         HR
11  12   33      M         IT
12  13   26      F    Finance
13  14   31      M         IT
14  15   38      F         HR
15  16   45      M    Finance
16  17   28      F         IT
17  18   50      M         HR
18  19   41      F    Finance
19  20   39      M         IT
```

```
# Randomly sample 5 rows from the dataset

random_sample = df.sample(n=5, random_state=42)
print("\nRandom Sample (5 rows):\n", random_sample)
```

```
Random Sample (5 rows):
     ID  Age Gender Department
0    1   22      M         HR
17  18   50      M         HR
15  16   45      M    Finance
1    2   25      F    Finance
8    9   36      M         HR
```

```
# Stratified sampling by Department - sample 2 rows from each department

stratified_sample = df.groupby('Department', group_keys=False).apply(lambda x:
x.sample(min(len(x), 2), random_state=42))
print("\nStratified Sample (2 rows per Department):\n", stratified_sample)
```

```
Stratified Sample (2 rows per Department):
    ID  Age Gender Department
1   2   25      F    Finance
```

```
4   5   46      F    Finance
0   1   22      M         HR
5   6   52      M         HR
2   3   47      M         IT
3   4   35      F         IT
```

# 5. Write a Python function to calculate the range of a dataset.

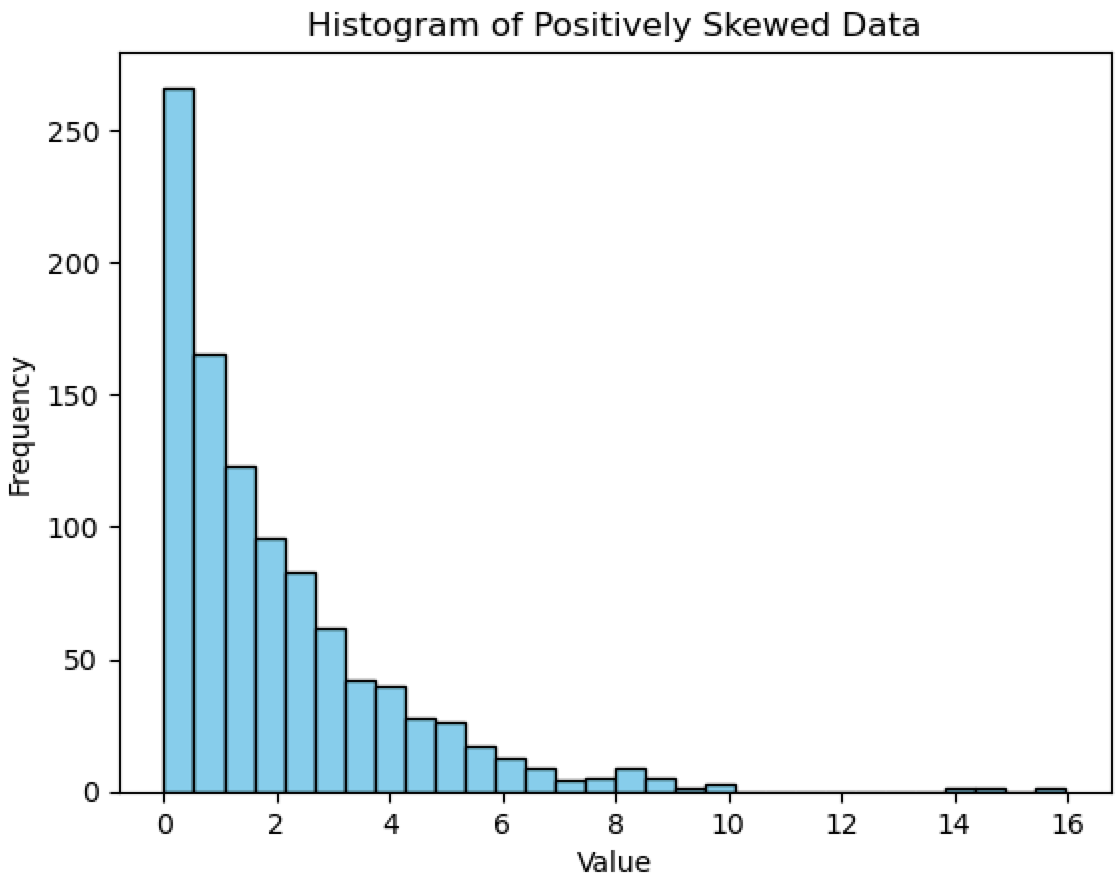
```
def calculate_range(data):
    if len(data) == 0:
        return None  # Return None if dataset is empty
    return max(data) - min(data)

# Example usage:
dataset = [10, 5, 8, 12, 7]
range_value = calculate_range(dataset)
print("Range of dataset:", range_value)
```

```
Range of dataset: 7
```

# 6. Create a dataset and plot its histogram to visualize skewness.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import skew

# Create a positively skewed dataset (e.g., exponential distribution)
data = np.random.exponential(scale=2, size=1000)

# Calculate skewness
data_skewness = skew(data)
print(f"Skewness of the dataset: {data_skewness:.2f}")

# Plot histogram
plt.hist(data, bins=30, color='skyblue', edgecolor='black')
plt.title('Histogram of Positively Skewed Data')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```

```
Skewness of the dataset: 1.97
```



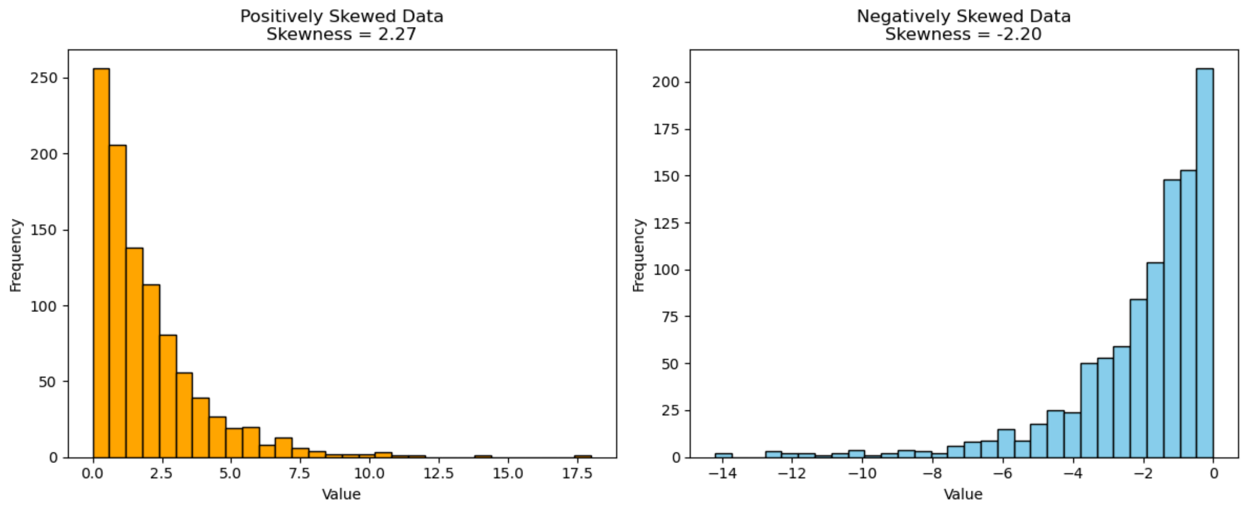
# 7. Calculate skewness and kurtosis of a dataset using Python libraries.

```
import numpy as np
from scipy.stats import skew, kurtosis

# Sample dataset
data = np.array([10, 12, 23, 23, 16, 23, 21, 16, 18, 19, 15])

# Calculate skewness
data_skewness = skew(data)

# Calculate kurtosis
data_kurtosis = kurtosis(data)  # By default, gives excess kurtosis

print(f"Skewness: {data_skewness:.3f}")
print(f"Kurtosis (excess): {data_kurtosis:.3f}")
```

```
Skewness: -0.299
Kurtosis (excess): -1.043
```

# 8. Generate a dataset and demonstrate positive and negative skewness.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import skew

# Generate positively skewed data (exponential distribution)
pos_skew_data = np.random.exponential(scale=2, size=1000)

# Generate negatively skewed data by negating exponential data
neg_skew_data = -np.random.exponential(scale=2, size=1000)

# Calculate skewness
pos_skewness = skew(pos_skew_data)
neg_skewness = skew(neg_skew_data)

# Plotting
fig, axs = plt.subplots(1, 2, figsize=(12, 5))

# Positive skew plot
axs[0].hist(pos_skew_data, bins=30, color='orange', edgecolor='black')
axs[0].set_title(f' Positively Skewed Data\nSkewness = {pos_skewness:.2f}')
axs[0].set_xlabel('Value')
axs[0].set_ylabel('Frequency')

# Negative skew plot
axs[1].hist(neg_skew_data, bins=30, color='skyblue', edgecolor='black')
axs[1].set_title(f' Negatively Skewed Data\nSkewness = {neg_skewness:.2f}')
axs[1].set_xlabel('Value')
axs[1].set_ylabel('Frequency')

plt.tight_layout()
plt.show()
```



# 9. Write a Python script to calculate covariance between two datasets.

```
import numpy as np

# Example datasets
x = [10, 20, 30, 40, 50]
y = [12, 24, 33, 45, 55]

# Calculate covariance matrix
cov_matrix = np.cov(x, y)

# Covariance between x and y is at position [0,1]
cov_xy = cov_matrix[0, 1]

print(f"Covariance between x and y: {cov_xy}")
```

```
Covariance between x and y: 267.5
```

# 10. Write a Python script to calculate the correlation coefficient between two datasets

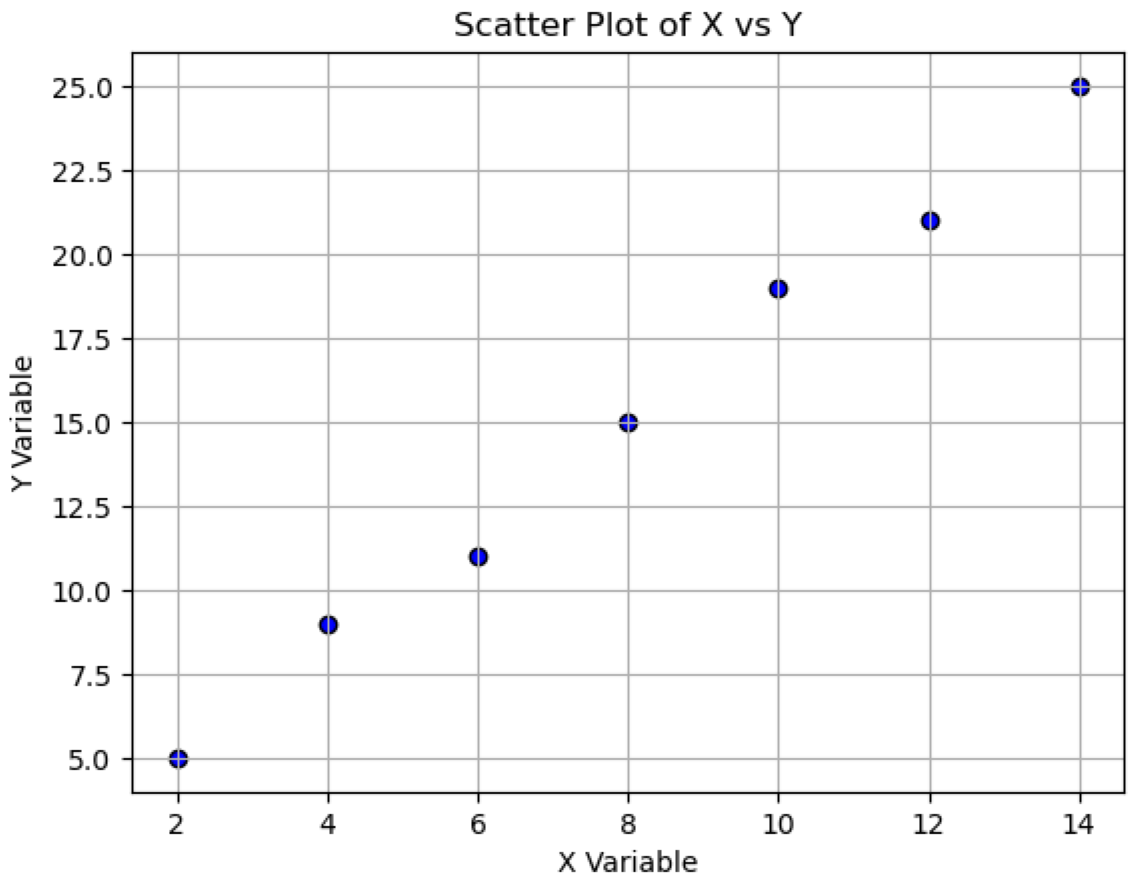
```
import numpy as np

# Example datasets
x = [10, 20, 30, 40, 50]
y = [12, 24, 33, 45, 55]
```

```
# Calculate correlation matrix
corr_matrix = np.corrcoef(x, y)

# Correlation coefficient between x and y is at position [0,1]
corr_xy = corr_matrix[0, 1]

print(f"Correlation coefficient between x and y: {corr_xy:.2f}")
```

```
Correlation coefficient between x and y: 1.00
```

# 11. Create a scatter plot to visualize the relationship between two variables.

```
import matplotlib.pyplot as plt

# Sample data
x = [2, 4, 6, 8, 10, 12, 14]
y = [5, 9, 11, 15, 19, 21, 25]

# Create scatter plot
plt.scatter(x, y, color='blue', edgecolor='black')

# Add titles and labels
plt.title('Scatter Plot of X vs Y')
plt.xlabel('X Variable')
plt.ylabel('Y Variable')

# Show plot
plt.grid(True)
plt.show()
```



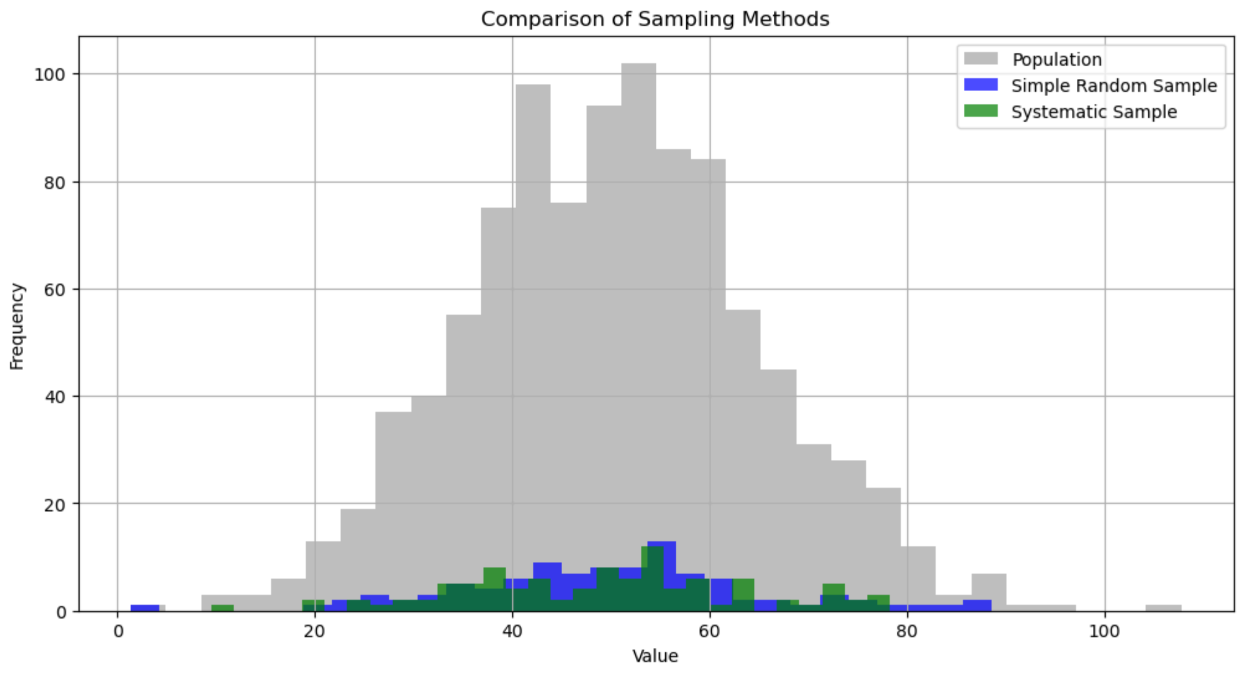
# 12. Implement and compare simple random sampling and systematic sampling.

1. Generate Population Data
We'll simulate a population of 1000 data points.

2. Perform Simple Random Sampling
Randomly select n samples from the population.

3. Perform Systematic Sampling
Pick every k-th item starting from a random index.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Create synthetic population data
np.random.seed(42)
population_size = 1000
population = pd.DataFrame({
    'ID': np.arange(1, population_size + 1),
    'Value': np.random.normal(loc=50, scale=15, size=population_size)  # Normal distribution
})

sample_size = 100  # Sample size for both methods

# Step 2: Simple Random Sampling
srs_sample = population.sample(n=sample_size, random_state=1)

# Step 3: Systematic Sampling
k = population_size // sample_size  # Sampling interval
start = np.random.randint(0, k)  # Random start point
systematic_indices = np.arange(start, population_size, k)
systematic_sample = population.iloc[systematic_indices]

# Step 4: Compare the two methods
print("SRS Sample Mean:", srs_sample['Value'].mean())
print("Systematic Sample Mean:", systematic_sample['Value'].mean())
print("Population Mean:", population['Value'].mean())

# Step 5: Visualization
plt.figure(figsize=(12, 6))
plt.hist(population['Value'], bins=30, alpha=0.5, label='Population', color='gray')
plt.hist(srs_sample['Value'], bins=30, alpha=0.7, label='Simple Random Sample', color='blue')
plt.hist(systematic_sample['Value'], bins=30, alpha=0.7, label='Systematic Sample',
color='green')
plt.legend()
plt.title("Comparison of Sampling Methods")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.grid(True)
plt.show()
```



Output Summary

- SRS: Purely random, good for avoiding bias, but may miss trends.

- Systematic: Easier to implement, but risky if population has a hidden periodic pattern.

# 13. Calculate the mean, median, and mode of grouped data.

To calculate the mean, median, and mode of grouped data, we need:

- Class intervals
- Frequencies for each class

Let's implement this in Python using a grouped frequency distribution.

Example Grouped Data:

Class Interval	Frequency
0 - 10	5
10 - 20	8
20 - 30	15
30 - 40	16
40 - 50	6

```
import numpy as np
from scipy import stats

# Example grouped data: class intervals and their frequencies
class_intervals = [(10, 20), (20, 30), (30, 40), (40, 50)]
frequencies = [5, 8, 12, 5]

# Calculate midpoints of each class interval
midpoints = [ (interval[0] + interval[1]) / 2 for interval in class_intervals ]

# Mean calculation
mean = np.average(midpoints, weights=frequencies)

# To calculate median, we find the class where cumulative frequency crosses half total
frequency
cum_freq = np.cumsum(frequencies)
n = sum(frequencies)
median_class_index = np.where(cum_freq >= n/2)[0][0]

# Median class info
L = class_intervals[median_class_index][0]  # lower class boundary
F = frequencies[median_class_index]
CF_prev = cum_freq[median_class_index - 1] if median_class_index > 0 else 0
class_width = class_intervals[0][1] - class_intervals[0][0]

# Median formula for grouped data
median = L + ((n/2 - CF_prev) / F) * class_width

# Mode calculation for grouped data (using formula)
f1 = frequencies[median_class_index]
f0 = frequencies[median_class_index - 1] if median_class_index > 0 else 0
f2 = frequencies[median_class_index + 1] if median_class_index < len(frequencies) - 1 else 0

mode = L + ((f1 - f0) / ((2*f1) - f0 - f2)) * class_width

print(f"Mean: {mean:.2f}")
print(f"Median: {median:.2f}")
print(f"Mode: {mode:.2f}")
```

```
Mean: 30.67
Median: 31.67
Mode: 33.64
```

# 14. Simulate data using Python and calculate its central tendency and dispersion.

Simulate a dataset (using normal distribution)

Calculate measures of central tendency:

- Mean

- Median

- Mode

Calculate measures of dispersion:

- Variance

- Standard deviation

- Range

- Interquartile Range (IQR)

```python
import numpy as np
import pandas as pd
from scipy import stats

# Step 1: Simulate data (e.g., exam scores)
np.random.seed(42)
data = np.random.normal(loc=70, scale=10, size=1000) # mean=70, std=10

# Step 2: Convert to DataFrame
df = pd.DataFrame(data, columns=['Scores'])

# Step 3: Central Tendency
mean = df['Scores'].mean()
median = df['Scores'].median()
mode = df['Scores'].mode()[0]

# Step 4: Dispersion
variance = df['Scores'].var()
```

```
std_dev = df['Scores'].std()
range_val = df['Scores'].max() - df['Scores'].min()
iqr = df['Scores'].quantile(0.75) - df['Scores'].quantile(0.25)

# Step 5: Print Results
print("Central Tendency Measures:")
print(f"Mean: {mean:.2f}")
print(f"Median: {median:.2f}")
print(f"Mode: {mode:.2f}")

print("\nDispersion Measures:")
print(f"Variance: {variance:.2f}")
print(f"Standard Deviation: {std_dev:.2f}")
print(f"Range: {range_val:.2f}")
print(f"IQR (Interquartile Range): {iqr:.2f}")
```

```
Central Tendency Measures:
Mean: 70.19
Median: 70.25
Mode: 37.59

Dispersion Measures:
Variance: 95.89
Standard Deviation: 9.79
Range: 70.94
IQR (Interquartile Range): 12.96
```

# 15. Use NumPy or pandas to summarize a dataset’s descriptive statistics.

Here is how to use NumPy and Pandas to summarize a dataset’s descriptive statistics such as:

- Count

- Mean

- Standard Deviation

- Min / Max

- Quartiles

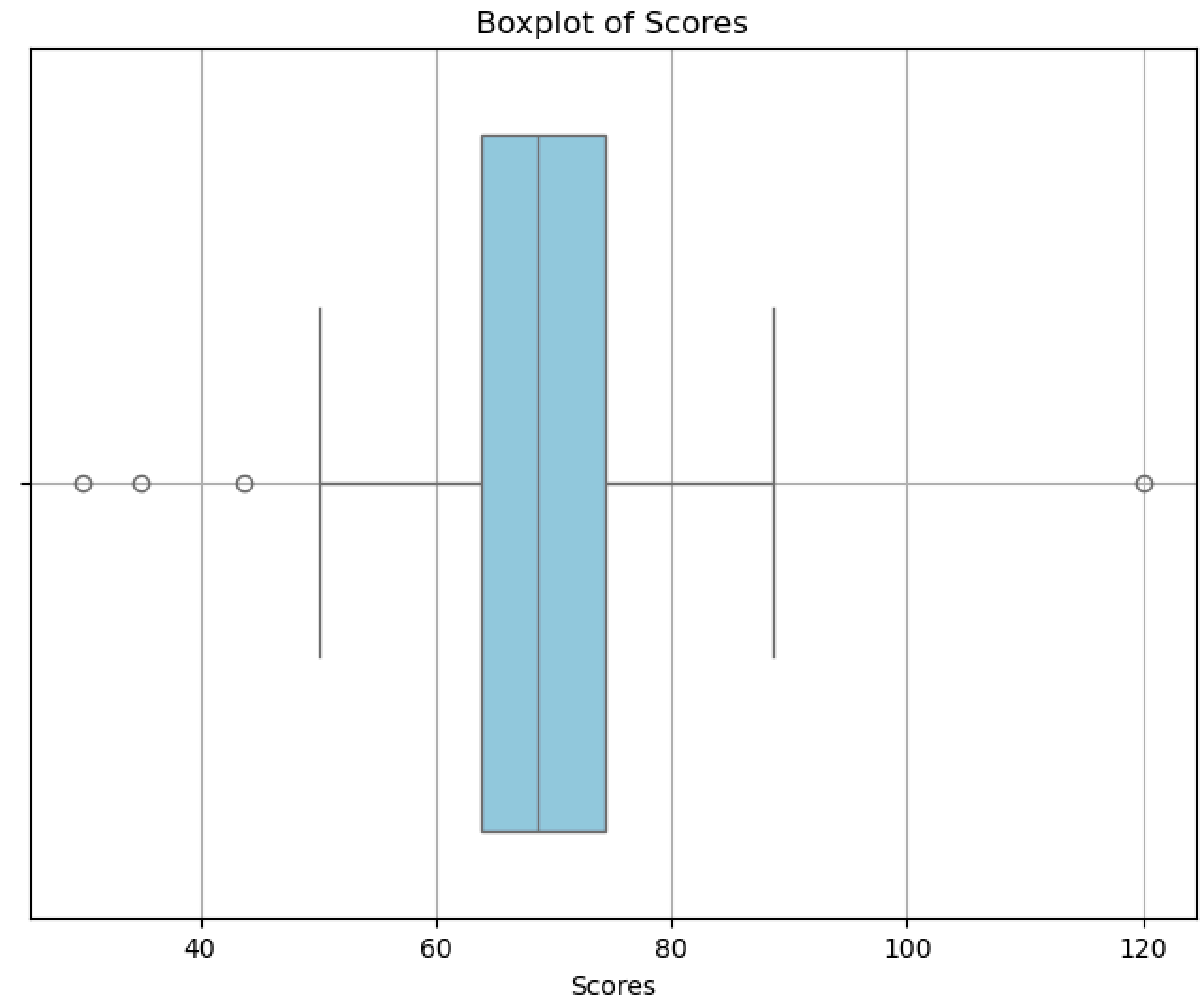
- Skewness / Kurtosis (optional)

```
import numpy as np
import pandas as pd

# Step 1: Simulate a dataset (e.g., student scores, ages, income)
np.random.seed(42)
data = {
  'Scores': np.random.normal(75, 10, 100),     # Normal distribution
  'Age': np.random.randint(18, 30, 100),       # Uniform distribution
  'Income': np.random.normal(50000, 8000, 100) # Income in dollars
}

df = pd.DataFrame(data)

# Step 2: Descriptive statistics using pandas
summary = df.describe()

# Step 3: Add skewness and kurtosis (optional)
summary.loc['skew'] = df.skew()
summary.loc['kurt'] = df.kurt()

# Step 4: Display summary
print("Descriptive Statistics Summary:\n")
print(summary)
```

```
Descriptive Statistics Summary:

           Scores         Age        Income
count  100.000000  100.000000    100.000000
mean    73.961535   23.110000  49020.187800
std      9.081684    3.722427   8680.076214
min     48.802549   18.000000  26964.344852
25%     68.990943   20.000000  42332.122135
50%     73.730437   23.000000  49044.764824
75%     79.059521   26.000000  54393.449480
max     93.522782   29.000000  68186.761264
skew    -0.177948    0.132322      0.030531
kurt    -0.100977   -1.303631     -0.482040
```

# 16. Plot a boxplot to understand the spread and identify outliers.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Simulate a dataset (e.g., test scores with some outliers)
np.random.seed(42)
scores = np.random.normal(loc=70, scale=10, size=100)
scores = np.append(scores, [30, 35, 120])  # Adding outliers

df = pd.DataFrame({'Scores': scores})

# Step 2: Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x=df['Scores'], color='skyblue')

plt.title("Boxplot of Scores")
plt.xlabel("Scores")
plt.grid(True)
plt.show()
```



What You will See:

- Box (IQR): Middle 50% of the data
- Line in box: Median
- Whiskers: 1.5×IQR range
- Dots beyond whiskers: Outliers

# 17. Calculate the interquartile range (IQR) of a dataset

The Interquartile Range (IQR) is a measure of statistical dispersion, calculated as:

IQR = Q3 - Q1

Where:
  Q1 is the 25th percentile (first quartile)
  Q3 is the 75th percentile (third quartile)

```
import numpy as np

# Example dataset
data = [7, 15, 36, 39, 40, 41, 42, 43, 47, 49]

# Calculate Q1 and Q3
Q1 = np.percentile(data, 25)
Q3 = np.percentile(data, 75)

# Calculate IQR
IQR = Q3 - Q1

print(f"Q1: {Q1}")
print(f"Q3: {Q3}")
print(f"IQR: {IQR}")
```

```
Q1: 36.75
Q3: 42.75
IQR: 6.0
```

# 18. Implement Z-score normalization and explain its significance.

Z-Score Normalization:

Z-score normalization, also known as standardization, is a technique used to scale data based on the mean and standard deviation of the dataset.

It transforms the data so that it has:

- A mean of 0

- A standard deviation of 1

The formula is:

$$Z = (X - \mu)/\sigma$$

Where:

- X is the original value
- μ is the mean of the dataset
- σ is the standard deviation of the dataset

Significance of Z-Score Normalization:

- Removes units and allows comparison between different scales
- Essential for algorithms that rely on distance (e.g., KNN, SVM, PCA)
- Helps achieve faster convergence in gradient-based models (e.g., logistic regression, neural networks)

```
import numpy as np

# Example dataset
data = [10, 20, 30, 40, 50]

# Convert to NumPy array
data = np.array(data)

# Calculate mean and standard deviation
mean = np.mean(data)
std = np.std(data)

# Z-score normalization
z_scores = (data - mean) / std

print("Original Data:", data)
print("Z-Score Normalized Data:", z_scores)
```

```
Original Data: [10 20 30 40 50]
Z-Score Normalized Data: [-1.41421356 -0.70710678  0.          0.70710678
1.41421356]
```

# 19. Compare two datasets using their standard deviations.

Standard deviation is a measure of how spread out the values in a dataset are. When comparing two datasets:

- A higher standard deviation means more variability (spread).

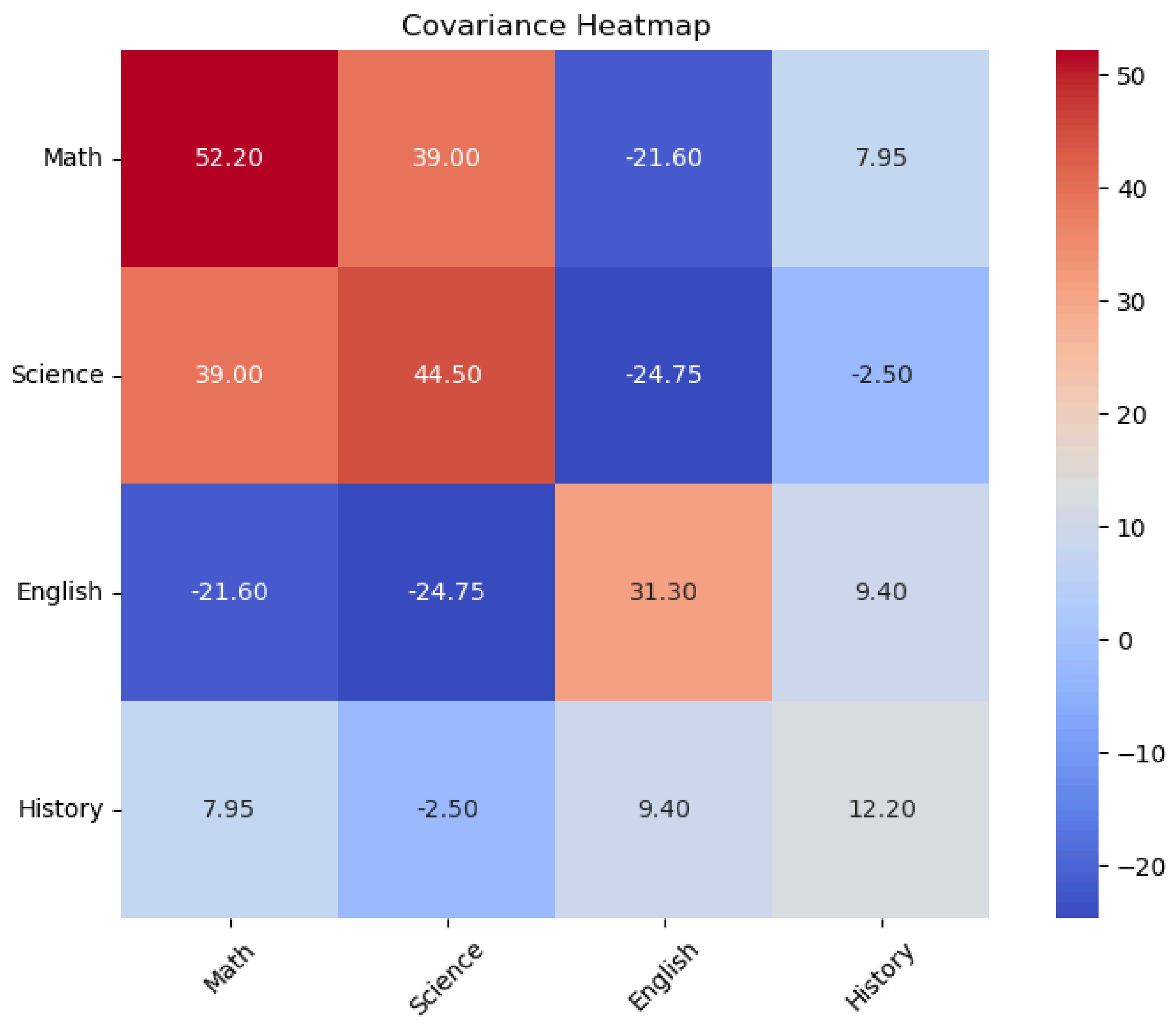
- A lower standard deviation means the data points are more closely clustered around the mean.

```
import numpy as np

# Example datasets
data1 = [10, 20, 30, 40, 50]
data2 = [25, 27, 29, 31, 33]

# Calculate standard deviations
std1 = np.std(data1)
std2 = np.std(data2)

# Print results
print("Standard Deviation of Dataset 1:", std1)
print("Standard Deviation of Dataset 2:", std2)

# Compare
if std1 > std2:
    print("Dataset 1 has more variability.")
elif std1 < std2:
    print("Dataset 2 has more variability.")
else:
    print("Both datasets have the same variability.")
```

```
Standard Deviation of Dataset 1: 14.142135623730951
Standard Deviation of Dataset 2: 2.8284271247461903
Dataset 1 has more variability.
```

# 20. Write a Python program to visualize covariance using a heatmap.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Create a sample DataFrame with multiple variables
data = {
    'Math': [88, 92, 80, 89, 100],
    'Science': [90, 85, 78, 92, 95],
    'English': [70, 78, 85, 80, 75],
    'History': [65, 70, 72, 68, 74]
}

df = pd.DataFrame(data)

# Compute the covariance matrix
cov_matrix = df.cov()

# Set the plot size and style
plt.figure(figsize=(8, 6))
sns.heatmap(cov_matrix, annot=True, cmap='coolwarm', fmt=".2f", square=True)

# Set plot title and labels
plt.title("Covariance Heatmap")
plt.xticks(rotation=45)
plt.yticks(rotation=0)

plt.tight_layout()
plt.show()
```



Notes:

- df.cov() calculates the covariance matrix.

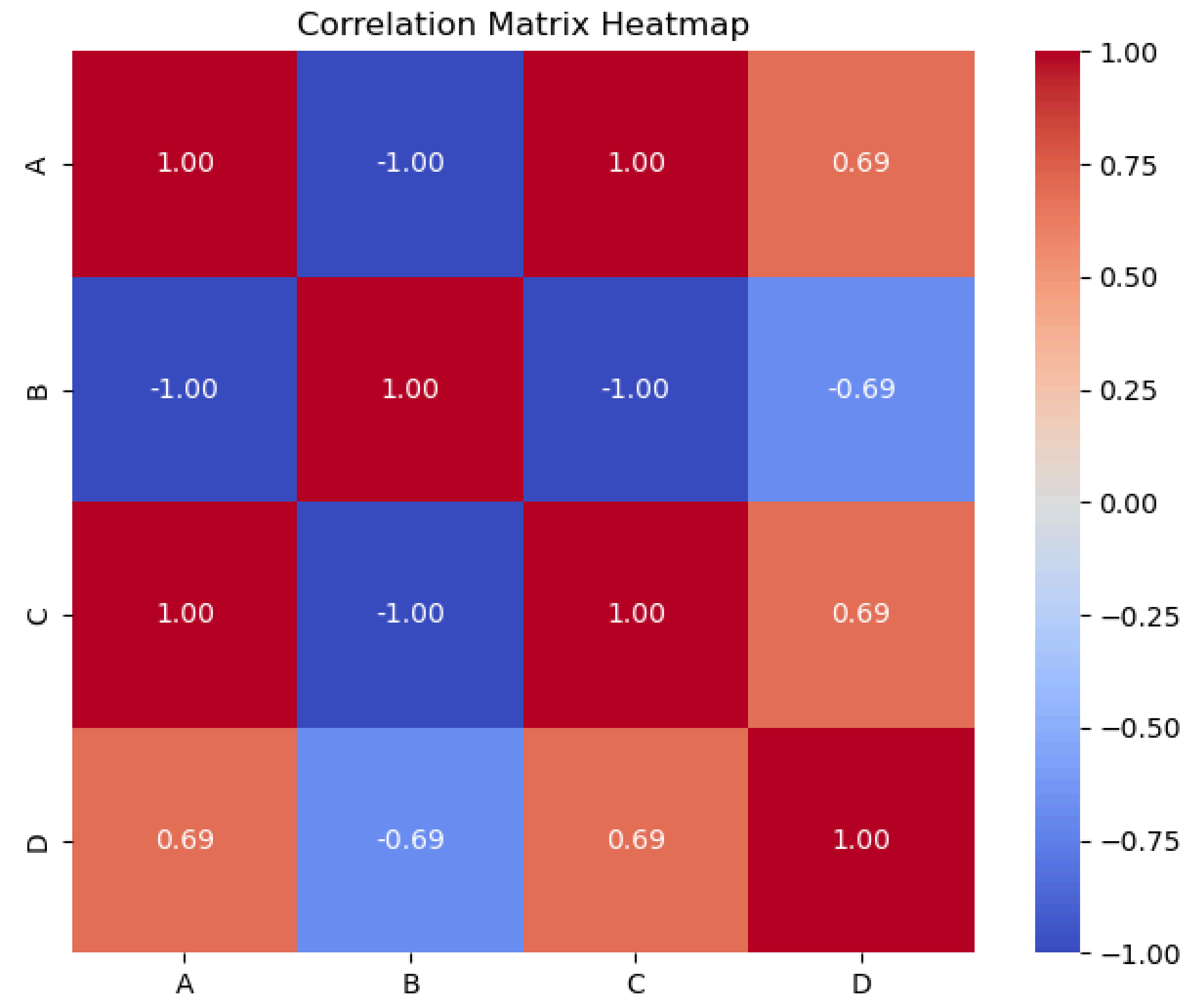
- seaborn.heatmap() is used to visualize it.

- Darker colors indicate stronger relationships.

# 21. Use seaborn to create a correlation matrix for a dataset.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Example dataset
data = {
    'A': [1, 2, 3, 4, 5],
    'B': [5, 4, 3, 2, 1],
    'C': [2, 3, 4, 5, 6],
    'D': [5, 7, 6, 8, 7]
}

df = pd.DataFrame(data)

# Calculate correlation matrix
corr_matrix = df.corr()

# Plot correlation matrix using seaborn heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", square=True)

plt.title('Correlation Matrix Heatmap')
plt.show()
```



This code:

- Creates a small dataset with four variables.
- Calculates the correlation matrix using .corr().
- Plots it as a heatmap with annotations and a cool-to-warm color scale.

# 22. Generate a dataset and implement both variance and standard deviation computations.

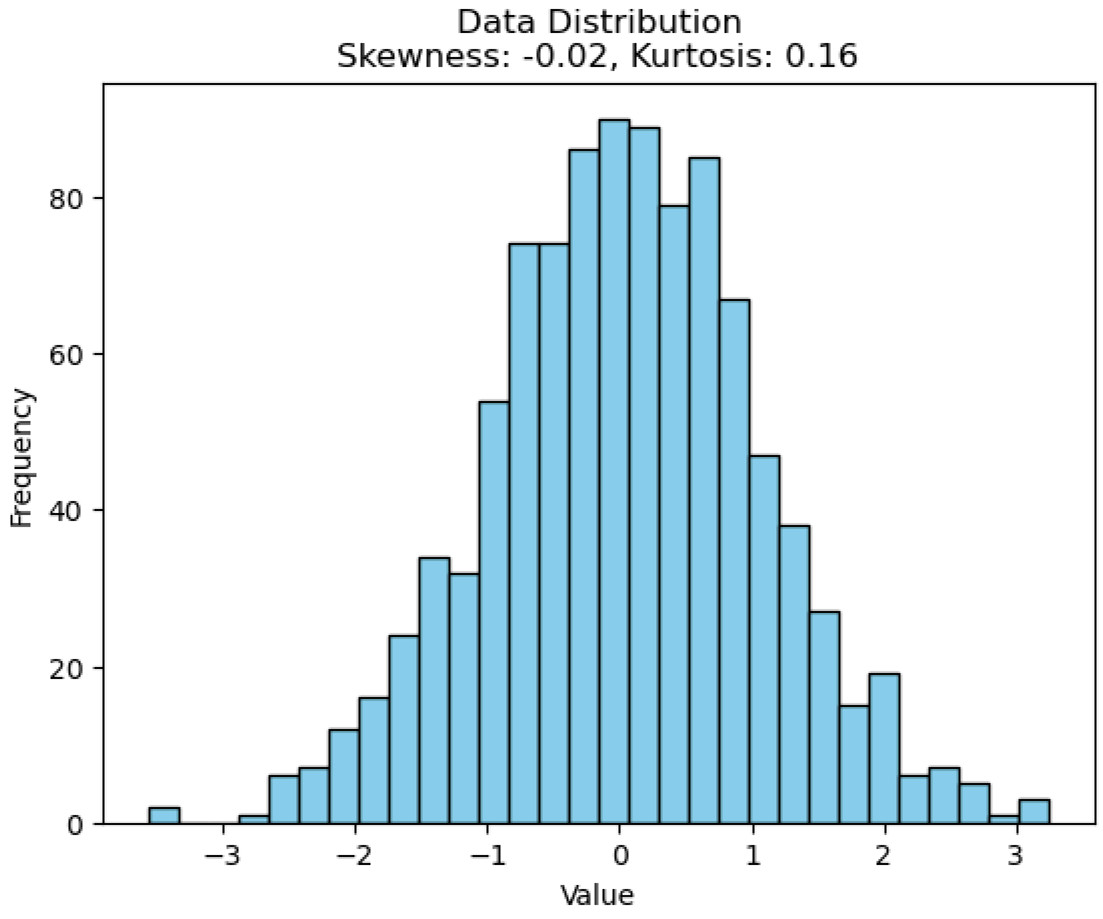
```
import numpy as np

# Generate a random dataset of 20 numbers between 0 and 100
np.random.seed(42)  # for reproducibility
```

```
data = np.random.randint(0, 100, size=20)

# Calculate variance and standard deviation
variance = np.var(data)
std_deviation = np.std(data)

print("Dataset:", data)
print(f"Variance: {variance:.2f}")
print(f"Standard Deviation: {std_deviation:.2f}")
```

```
Dataset: [51 92 14 71 60 20 82 86 74 74 87 99 23  2 21 52  1 87 29 37]
Variance: 990.49
Standard Deviation: 31.47
```

# 23. Visualize skewness and kurtosis using Python libraries like matplotlib or seaborn

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import skew, kurtosis

# Generate sample data
data = np.random.normal(loc=0, scale=1, size=1000)

# Calculate skewness and kurtosis
data_skewness = skew(data)
data_kurtosis = kurtosis(data)

# Plot histogram
plt.hist(data, bins=30, color='skyblue', edgecolor='black')
plt.title(f'Data Distribution\nSkewness: {data_skewness:.2f}, Kurtosis: {data_kurtosis:.2f}')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```



# 24. Implement the Pearson and Spearman correlation coefficients for a dataset.

```
import numpy as np
from scipy.stats import pearsonr, spearmanr

# Sample data
x = [10, 20, 30, 40, 50]
y = [12, 24, 33, 45, 55]

# Calculate Pearson correlation
pearson_corr, _ = pearsonr(x, y)

# Calculate Spearman correlation
spearman_corr, _ = spearmanr(x, y)

print(f"Pearson correlation coefficient: {pearson_corr:.3f}")
print(f"Spearman correlation coefficient: {spearman_corr:.3f}")
```

```
Pearson correlation coefficient: 0.999
Spearman correlation coefficient: 1.000
```