```
## **Statistics Basics (Module 1)**

**Practical Questions - Part 2**
```

Statistics Basics (Module 1)

Practical Questions - Part 2

Q22. Generate a dataset and implement both variance and standard deviation computations.

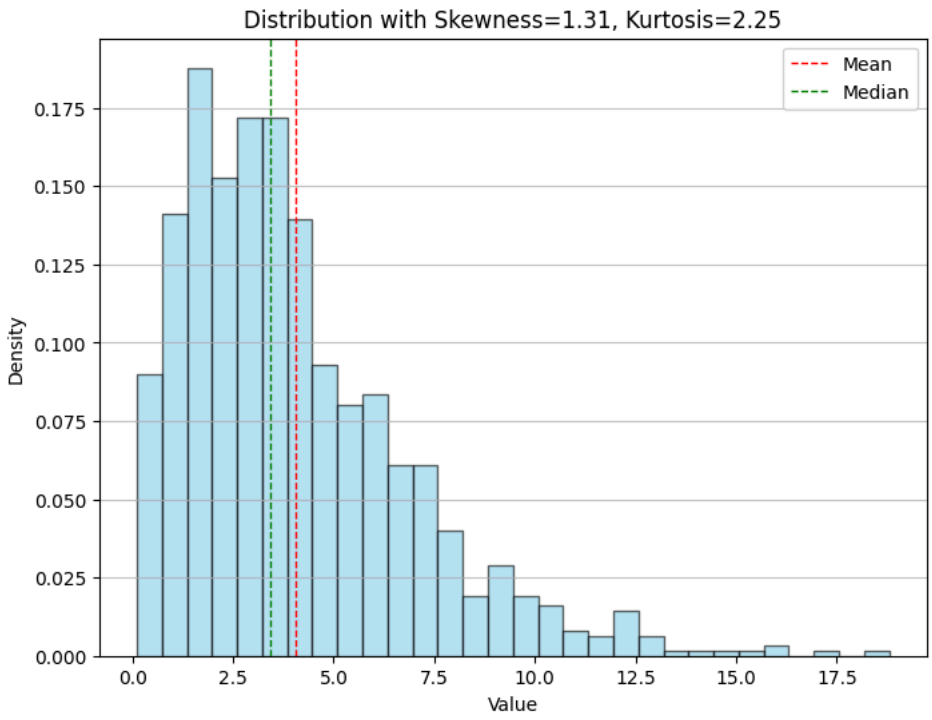
```
# prompt: Generate a dataset and implement both variance and standard deviation computations.

import numpy as np

def calculate_variance_std(data):
    """Calculates the variance and standard deviation of a dataset."""
    variance = np.var(data)
    std_dev = np.std(data)
    return variance, std_dev

# Generate a sample dataset
data = np.random.rand(100)  # Example: 100 random numbers between 0 and 1

# Calculate variance and standard deviation
variance, std_dev = calculate_variance_std(data)

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

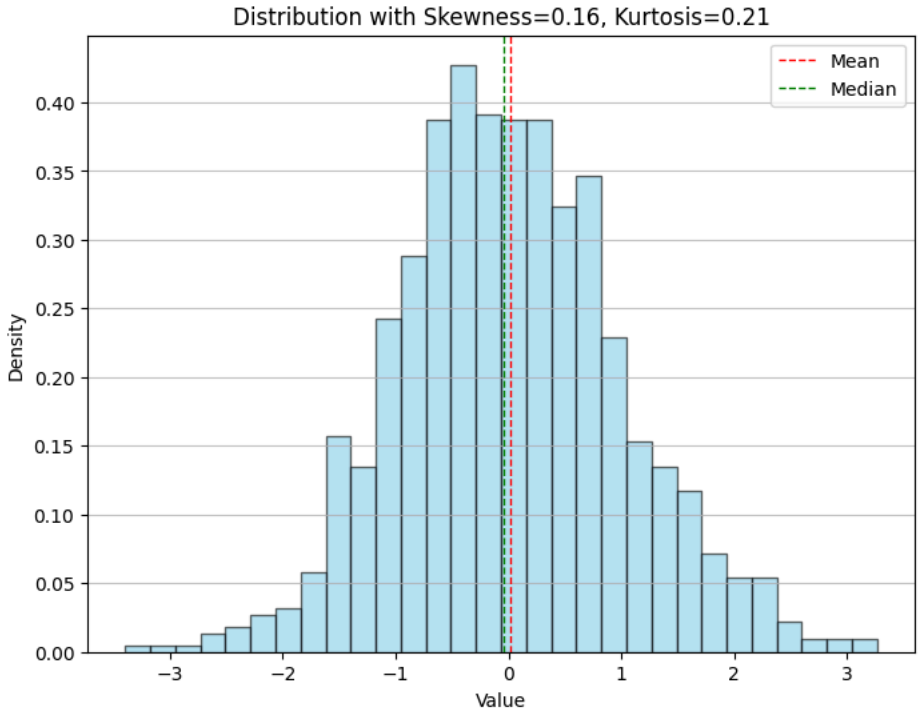
```
Variance: 0.07510221108737988
Standard Deviation: 0.27404782627742164
```

Q23. Visualize skewness and kurtosis using Python libraries like matplotlib or seaborn.

```
# prompt: Visualize skewness and kurtosis using Python libraries like matplotlib or seaborn.

import matplotlib.pyplot as plt
import numpy as np
import scipy.stats as stats

def visualize_skewness_kurtosis(data):
    """Visualizes skewness and kurtosis of a dataset.

    Args:
        data: A list or numpy array of numerical values.
    """
    try:
        # Calculate skewness and kurtosis
        skewness = stats.skew(data)
        kurtosis = stats.kurtosis(data)

        # Create the plot
        plt.figure(figsize=(8, 6))
        plt.hist(data, bins=30, density=True, alpha=0.6, color='skyblue', edgecolor='black')
        plt.title(f"Distribution with Skewness={skewness:.2f}, Kurtosis={kurtosis:.2f}")
        plt.xlabel("Value")
        plt.ylabel("Density")

        # Add vertical lines for mean, median
        plt.axvline(np.mean(data), color='red', linestyle='dashed', linewidth=1, label="Mean")
        plt.axvline(np.median(data), color='green', linestyle='dashed', linewidth=1, label="Median")
        plt.legend()

        plt.grid(axis='y', alpha=0.75)
        plt.show()

    except Exception as e:
        print(f"An error occurred: {e}")

# Example usage
data = np.random.gamma(2, 2, 1000) # Example: Positively skewed data
visualize_skewness_kurtosis(data)

data = np.random.normal(0,1, 1000) # Example: Normal Distribution
```

```
visualize_skewness_kurtosis(data)
```





Q24. Implement the Pearson and Spearman correlation coefficients for a dataset.

```
# prompt: Implement the Pearson and Spearman correlation coefficients for a dataset.

import numpy as np

def pearson_correlation(dataset1, dataset2):
    """
    Calculates the Pearson correlation coefficient between two datasets.
    """
    if len(dataset1) != len(dataset2):
        raise ValueError("Datasets must have the same length.")

    dataset1 = np.array(dataset1)
    dataset2 = np.array(dataset2)

    return np.corrcoef(dataset1, dataset2)[0, 1]
```

```
def spearman_correlation(dataset1, dataset2):
    """
    Calculates the Spearman rank correlation coefficient between two datasets.
    """
    if len(dataset1) != len(dataset2):
        raise ValueError("Datasets must have the same length.")

    dataset1 = np.array(dataset1)
    dataset2 = np.array(dataset2)

    return stats.spearmanr(dataset1, dataset2)[0]


# Example usage
data1 = [1, 2, 3, 4, 5]
data2 = [5, 4, 3, 2, 1]
data3 = [1,2,3,4,5]

pearson_coeff = pearson_correlation(data1, data2)
spearman_coeff = spearman_correlation(data1, data2)
pearson_coeff2 = pearson_correlation(data1, data3)
spearman_coeff2 = spearman_correlation(data1, data3)

print(f"Pearson Correlation Coefficient: {pearson_coeff}")
print(f"Spearman Correlation Coefficient: {spearman_coeff}")
print(f"Pearson Correlation Coefficient: {pearson_coeff2}")
print(f"Spearman Correlation Coefficient: {spearman_coeff2}")
```

```
Pearson Correlation Coefficient: -0.9999999999999999
Spearman Correlation Coefficient: -0.9999999999999999
Pearson Correlation Coefficient: 0.9999999999999999
Spearman Correlation Coefficient: 0.9999999999999999
```

End of Assignment