**PROGRAM 2: Study of Python Basic Libraries such as Statistics, Math, Numpy and Scipy**

**1. Statistics Library:**

The Statistics library in Python provides functions for calculating mathematical statistics of numeric data. Here are some of the key functions:

mean(): Returns the mean (average) of the given data.

median(): Returns the median (middle value) of the given data.

mode(): Returns the most frequently occurring value in the given data.

stdev(): Returns the standard deviation of the given data.

variance(): Returns the variance of the given data.

median\_low(): Returns the low median of the given data.

median\_high(): Returns the high median of the given data.

pstdev(): Returns the population standard deviation of the given data.

pvariance(): Returns the population variance of the given data

**Program:**

import statistics

data = [1, 2, 3, 4, 5]

print("Mean:", statistics.mean(data))

print("Median:", statistics.median(data))

print("Mode:", statistics.mode(data))

print("Standard Deviation:", statistics.stdev(data))

print("Variance:", statistics.variance(data))

**Output:**

Mean: 3

Median: 3

Mode: 1

Standard Deviation: 1.5811388300841898

Variance: 2.5

**2. Math Library:**

The Math library in Python provides functions for various mathematical operations. Here are some of the key functions:

sin(): Returns the sine of the given angle in radians.

cos(): Returns the cosine of the given angle in radians.

tan(): Returns the tangent of the given angle in radians.

exp(): Returns the exponential of the given number.

log(): Returns the natural logarithm of the given number.

sqrt(): Returns the square root of the given number.

pow(): Returns the value of the first argument raised to the power of the second argument.

ceil(): Returns the ceiling of the given number (the smallest integer not less than the given number).

floor(): Returns the floor of the given number (the largest integer not greater than the given number).

**Program:**

import math

print("Sine of 30 degrees:", math.sin(math.radians(30)))

print("Cosine of 60 degrees:", math.cos(math.radians(60)))

print("Tangent of 45 degrees:", math.tan(math.radians(45)))

print("Exponential of 2:", math.exp(2))

print("Natural Logarithm of 10:", math.log(10))

print("Square Root of 16:", math.sqrt(16))

print("2 raised to the power of 3:", math.pow(2, 3))

print("Ceiling of 4.7:", math.ceil(4.7))

print("Floor of 4.7:", math.floor(4.7))

**Output:**

Sine of 30 degrees: 0.49999999999999994

Cosine of 60 degrees: 0.5000000000000001

Tangent of 45 degrees: 0.9999999999999999

Exponential of 2: 7.38905609893065

Natural Logarithm of 10: 2.302585092994046

Square Root of 16: 4.0

2 raised to the power of 3: 8.0

Ceiling of 4.7: 5

Floor of 4.7: 4

**3. NumPy Library**

NumPy (Numerical Python) is a library for working with arrays and mathematical operations. Here are some of the key features:

Arrays: NumPy provides support for large, multi-dimensional arrays and matrices.

Vectorized Operations: NumPy provides vectorized operations for performing operations on entire arrays at once.

Linear Algebra: NumPy provides functions for linear algebra operations such as matrix multiplication and eigenvalue decomposition

**Program:**

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

print("Array + 2:", arr + 2)

print("Array \* 2:", arr \* 2)

print("Matrix Multiplication:", np.dot(arr, arr.T))

print("Eigenvalue Decomposition:", np.linalg.eigvals(arr))

**Output:**

Array + 2: [[3 4 5]

[6 7 8]]

Array \* 2: [[ 2 4 6]

[ 8 10 12]]

Matrix Multiplication: [[14 32]

[32 77]]

**4. SciPy Library**

SciPy (Scientific Python) is a library for scientific and engineering applications. Here are some of the key features:

Signal Processing: SciPy provides functions for signal processing such as filtering and convolution.

Linear Algebra: SciPy provides functions for linear algebra operations such as eigenvalue decomposition and singular value decomposition.

Optimization: SciPy provides functions for optimization such as minimization and maximization.

Statistics: SciPy provides functions for statistical analysis such as hypothesis testing and confidence intervals.

**Program**:

import numpy as np

from scipy import signal

t = np.linspace(0, 1, 100)

x = np.sin(2 \* np.pi \* 10 \* t) + 0.5 \* np.sin(2 \* np.pi \* 20 \* t)

b, a = signal.butter(4, 0.125)

y = signal.filtfilt(b, a, x)

import matplotlib.pyplot as plt

plt.plot(t, x, label='Original signal')

plt.plot(t, y, label='Filtered signal')

plt.legend()

plt.show()

**Output:**

