Python Programming

Lab:- 29(Scipy Cluster & Constant)

Student Id:- AF0417098

Student Name:- Ankush

Introduction to SciPy

SciPy is a powerful Python library used for scientific and technical computing. It builds on top of NumPy and provides a wide range of functions and tools for mathematical computations, optimization, integration, interpolation, eigenvalue problems, and signal processing, among many other tasks.

SciPy is essential for data scientists, engineers, and researchers who require advanced computational methods. It is widely used for solving problems in scientific computing, mathematics, engineering, and machine learning.

1. SciPy vs. NumPy

SciPy builds upon NumPy, which is the fundamental package for array operations in Python. NumPy handles basic array operations like matrix and vector multiplication, while SciPy provides higher-level functions to perform operations such as optimization, signal processing, and solving differential equations.

- NumPy is mainly for numerical operations with arrays and matrices.
- SciPy extends NumPy and provides a collection of mathematical algorithms and convenience functions that make scientific computing easier.

2. SciPy Ecosystem

SciPy is part of the broader SciPy ecosystem which includes libraries like:

- NumPy: For numerical operations and array manipulation.
- Matplotlib: For data visualization.
- Pandas: For data manipulation and analysis.
- SymPy: For symbolic computation.
- scikit-learn: For machine learning.

3. Installation of SciPy

To install SciPy, you can use Python's package manager, pip:

```
pip install scipy
```

Once installed, you can import the library into your Python script:

```
import scipy
```

4. SciPy Sub-packages

SciPy is organized into sub-packages, each focusing on a specific scientific computing domain. Here are some of the key sub-packages:

- scipy.integrate: For integration (numerical integration, solving ordinary differential equations).
- scipy.optimize: For optimization and root finding (finding minimums and maximums).
- scipy.linalg: For linear algebra operations (matrix decompositions, solving systems of linear equations).
- scipy.fftpack: For Fourier transforms (used in signal processing and other areas).
- scipy.signal: For signal processing (filtering, convolution, etc.).

- **scipy.spatial:** For working with spatial data structures and algorithms (KD-trees, distance computations).
- **scipy.stats**: For statistical functions (probability distributions, hypothesis testing)

5.Real-World Applications of SciPy

- SciPy is versatile and widely used in several fields:
- **Engineering:** Used for solving differential equations, system modeling, signal processing, etc.
- **Physics:** For simulations, modeling dynamic systems, and solving physical equations.
- **Machine Learning:** SciPy is often used in feature extraction and data processing stages of machine learning pipelines.
- **Finance:** Optimization algorithms help solve financial modeling problems, portfolio optimization, and risk management.

6. Why Use SciPy?

- Ease of Use: SciPy provides high-level functions for complex mathematical operations, making it accessible even for non-experts in numerical computing.
- Performance: SciPy is optimized for performance, leveraging compiled libraries such as BLAS, LAPACK, and Fortran.
- Comprehensive: It has an extensive range of functions that cover most aspects of scientific computing.

Assignment Questions:-

Ques1:- Convert inches to centimeter.

https://raw.githubusercontent.com/AnudipAE/DANLC/master/people_heights.csv

Program:-

```python
# Convert inches to centimeter.
# https://raw.githubusercontent.com/AnudipAE/DANLC/master/people_heights.csv


import pandas as pd

# Load the data and convert inches to centimeters
df = pd.read_csv("https://raw.githubusercontent.com/AnudipAE/DANLC/master/people_heights.csv")
df['Height (cm)'] = df['Height (inches)'] * 2.54
print(df)

```

Output:-

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  TEST RESULTS  PORTS
PS C:\Users\Raj Kumar\Desktop\python programming> & "C:/Users/Raj Kumar/AppData/Local/Programs/Python/Python312/python
/Users/Raj Kumar/Desktop/python programming/lab29.py"
          Name  Height (inches)  Height (cm)
0     Person 1            60.03     152.4762
1     Person 2            49.51     125.7554
2     Person 3            82.97     210.7438
3     Person 4            64.19     163.0426
4     Person 5            54.42     138.2268
..         ...              ...          ...
95   Person 96            76.69     194.7926
96   Person 97            68.06     172.8724
97   Person 98            57.89     147.0406
98   Person 99            63.56     161.4424
99  Person 100            81.85     207.8990

[100 rows x 3 columns]
PS C:\Users\Raj Kumar\Desktop\python programming>
```

Ques 2. Convert Giga Byte to Mega Byte.

https://raw.githubusercontent.com/AnudipAE/DANLC/master/file_size.csv

Program:-

```
C:\Users\Raj Kumar\Desktop\python programming\lab21.py
 6
 7    # # Load the data and convert inches to centimeters
 8    # df = pd.read_csv("https://raw.githubusercontent.com/AnudipAE/DANLC/master/people_heights.csv")
 9    # df['Height (cm)'] = df['Height (inches)'] * 2.54
10    # print(df)
11
12    #2.
13    # Load the data and convert GB to MB
14    df = pd.read_csv("https://raw.githubusercontent.com/AnudipAE/DANLC/master/file_size.csv")
15    df['Size (MB)'] = df['Size (GB)'] * 1024
16    print(df)
17
18
```

Output:-

```
[100 rows x 3 columns]
PS C:\Users\Raj Kumar\Desktop\python programming> & "C:/Users/Raj Kumar/AppData/Local/Programs/Python/Python312/python.exe" "c
/Users/Raj Kumar/Desktop/python programming/lab29.py"
       Filename  Size (GB)  Size (MB)
0     file_1.txt       9.72    9953.28
1     file_2.txt       9.81   10045.44
2     file_3.txt       5.61    5744.64
3     file_4.txt       4.58    4689.92
4     file_5.txt       5.52    5652.48
..           ...        ...        ...
95   file_96.txt       1.29    1320.96
96   file_97.txt       7.11    7280.64
97   file_98.txt       4.86    4976.64
98   file_99.txt       7.89    8079.36
99  file_100.txt       5.52    5652.48

[100 rows x 3 columns]
```