**Unit 4: **

4.1. Introduction to Regular Expressions (REs)

Regular Expressions (REs) are sequences of characters that form search patterns. They are used for string matching, searching, and manipulation. Common applications include validating input, searching for patterns in text, and replacing text.

Syntax:

Regular expressions use a combination of literal characters and special characters (also called metacharacters) to define search patterns.

4.2. Importance of Regular Expressions

Text Processing:

REs are powerful tools for text processing tasks like finding specific patterns in large datasets, validating user input (e.g., email addresses), and text extraction.

Flexibility:

REs allow complex string manipulations using simple and concise syntax.

4.3. The `re` Module in Python

Python's `re` module provides functions to work with regular expressions. It allows pattern matching, searching, and text manipulation.

Basic Functions:

`re.search(pattern, string)`

- Searches for the first occurrence of the pattern in the string.
- Returns a match object if a match is found, otherwise returns `None`.
- Example:

  ```python
  import re
  match = re.search(r'\bword\b', 'This is a word in a sentence.')
  ```

```
print(match.group())  # Output: word
```

re.match(pattern, string)`

- Checks for a match only at the beginning of the string.
- Example:

```python
import re
match = re.match(r'Hello', 'Hello World!')
print(match.group())  # Output: Hello
```

-`re.findall(pattern, string)`

- Finds all non-overlapping matches of the pattern in the string.
- Returns a list of all matches.
- Example:

```python
import re
matches = re.findall(r'\d+', '123 abc 456 def 789')
print(matches)  # Output: ['123', '456', '789']
```

`re.sub(pattern, repl, string)`

- Replaces occurrences of the pattern in the string with a replacement string.
- Example:

```python
import re
result = re.sub(r'cat', 'dog', 'The cat is on the mat.')
print(result)  # Output: The dog is on the mat.
```

Regular expressions are powerful tools for text processing, providing a flexible way to search, match, and manipulate strings. Mastery of REs allows you to handle complex text-processing tasks efficiently, making them a valuable skill in Python programming. Understanding the syntax

and functions provided by the `re` module is essential for leveraging the full potential of regular expressions in your Python projects.

4.4 Python PyLab Module

While performing some tasks, we have to use graphs like line charts, bar graphs, etc., for many reasons like to make the task more interactive, to pass the information in a very interesting way, graphs are easy and self-explanatory, etc. That's why plotting a graph or chart is a very important and integrated part of many functions. Graphs and charts play a very important role in the field of programming, and developers are always recommended for using graphs in their programs. Therefore, it becomes very important that we should be aware of how we can plot graphs from a program. MATLAB is considered the best to plot graphs and charts, but it is not possible for everyone to use MATLAB for plotting graphs & charts. We have many interactive modules present in Python that allow us to plot graphs and charts in the output, but here we will talk about the module, which provides us a MATLAB-like namespace by importing functions. We will talk about the PyLab Module of Python in this tutorial, and we will learn how we can plot graphs and chart in the output using functions of this module in a program.

4.2.2 PyLab Module: Installation

As we have already discussed, the PyLab Module gets installed alongside the installation of the Matplotlib package. Still, if we want to use this module in a Python program, we should make sure that Matplotlib Module is present in our system. If Matplotlib is not present in the system, then we can use the following pip installer command in the command prompt terminal shell to install Matplotlib Module to get the PyLab Module with it:

```
pip install matplotlib
```

Other than this, we have also discussed that PyLab Module also uses the mathematical and vector operation functions from the Numpy Module. Therefore, we have to make sure that Numpy Module is also present in our system, and if it is not installed, then we can use the following command to install the numpy module from the command prompt terminal:

```
pip install numpy
```

4.5 Basic plotting with PyLab Module:

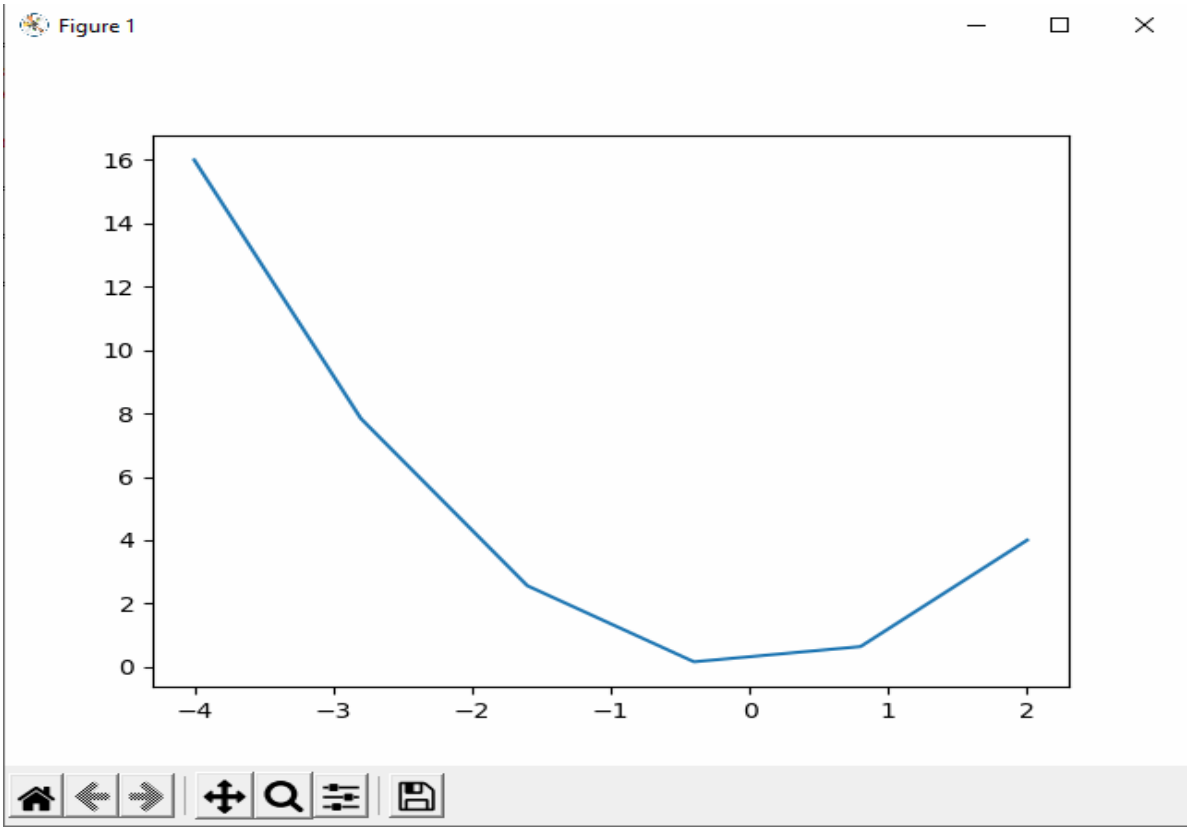
Plotting curves using the PyLab Module in a Python program requires the use of a plot command, and this command takes a pair of arrays or sequences having the same length. Let's understand this implementation through an example program.

Example:

Look at the following Python program from which we will plot a basic curve in the output:

```
Importing from Numpy
from numpy import *
# Importing from PyLab
from pylab import *
# X-axis of the curve
a = linspace(-4, 2, 6)
# Y-axis of the curve
b = a**2
# Plotting the curve with x and y-axis
plot(a, b)

# Showing curve in the output
show()
```



We have first imported all the functions from the Numpy and PyLab Module to use them in the program for plotting a curve in the output. After that, we have defined the X and Y-axis for the curve in the a & b variable, respectively. Then, we used the plot() function and provided a and b variables as arguments. In last, we used the show() function to display the curve we plotted in the output.

As we can see in the output that the curve is displayed with the X and Y-axis points, we have defined inside 'a' and 'b' variables, respectively.

4.6 Networking and Sockets

Networking refers to the communication between multiple computers or devices. In Python, networking can be achieved through sockets, which allow programs to communicate over a network, such as the internet or a local network.

4.7 What is a Socket?

A socket is an endpoint for sending and receiving data across a network. It allows for communication between two nodes in a network.

Sockets can use different protocols, with the most common being TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

TCP is reliable, ensuring data is received in the correct order, while UDP is faster but does not guarantee order or delivery.

4.8 Types of Sockets

- Stream Sockets (TCP):

TCP sockets are used for connection-oriented communication. They establish a reliable connection between the client and the server before data is transferred.

Characteristics:

- Reliable data transfer.
- Orderly delivery of data.
- Connection-based communication.

- Datagram Sockets (UDP):

UDP sockets are used for connectionless communication, where data packets are sent independently without establishing a connection.

Characteristics:

- Faster but less reliable.
- Data packets may arrive out of order.
- No connection establishment required.

4.9 Importing the Socket Module:**

- Python's `socket` module provides the necessary functions to create and work with sockets.

```python
import socket
```

4.10 Multithreading in Socket Programming

Multithreading allows a program to handle multiple tasks concurrently. In the context of socket programming, multithreading enables a server to handle multiple clients simultaneously.

Creating Threads:

- The `threading` module in Python provides tools for working with threads.

```python
import threading
```

Sockets are a fundamental aspect of networking, allowing Python programs to communicate across networks. Understanding socket programming is essential for developing networked applications such as web servers, chat applications, and more. By combining sockets with multithreading, you can create efficient and scalable networked systems capable of handling multiple clients concurrently.