**Shashwat Tripathi**

**D10A 58**

**BATCH: C**

**PYTHON LAB**

**Assignment No: 6**

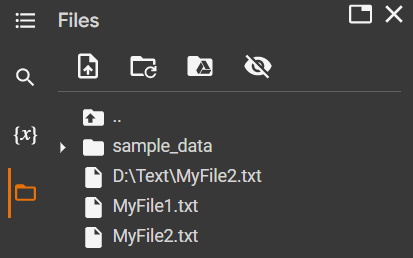
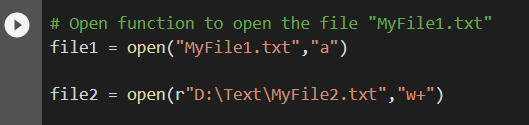
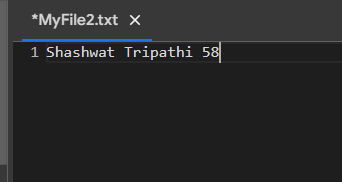
Aim: Write a python program to perform the following -

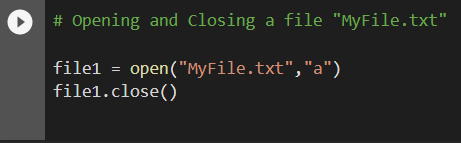
1. read and write operations on a file.

2. copy the contents of a file to another file.

3. count frequency of characters in a given file.

**Opening a file**



**Read and write from a file**

**# Program to show various ways to read and write data in a file.**

**file1 = open("myfile.txt","w")**

**L = ["This is Delhi \n","This is Paris \n","This is London \n"]**

**# \n is placed to indicate EOL (End of Line)**

**file1.write("Hello \n")**

**file1.writelines(L)**

**file1.close() #to change file access modes**

**file1 = open("myfile.txt","r+")**

**print("Output of Read function is ")**

**print(file1.read())**

**print()**

**# seek(n) takes the file handle to the nth byte from the beginning.**

**file1.seek(0)**

**print( "Output of Readline function is ")**

**print(file1.readline())**

**print()**

**file1.seek(0)**

**# To show difference between read and readline**

**print("Output of Read(9) function is ")**

**print(file1.read(9))**

**print()**

**file1.seek(0)**

**print("Output of Readline(9) function is ")**

**print(file1.readline(9))**

**file1.seek(0)**

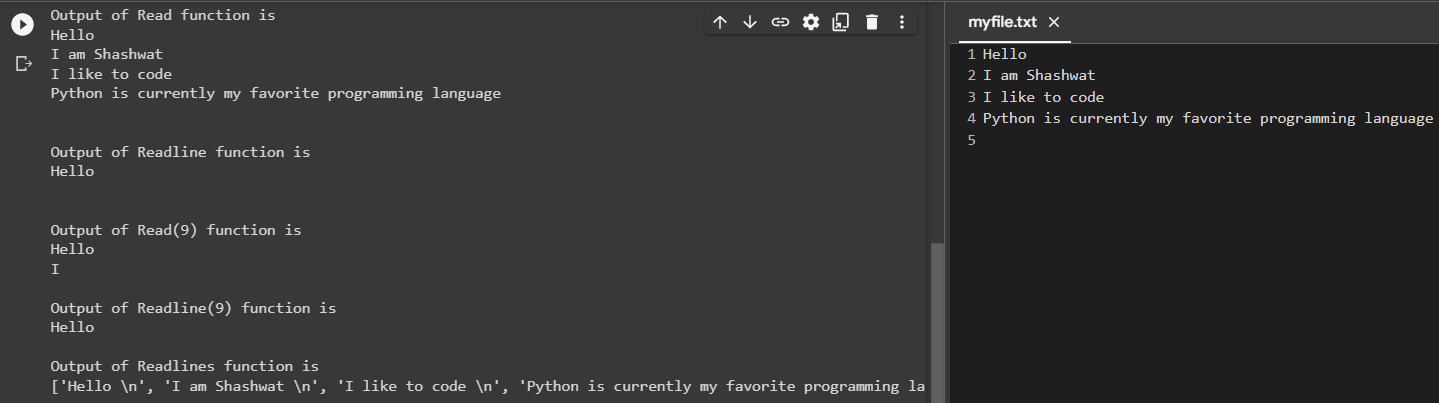
**# readlines function**

**print("Output of Readlines function is ")**

**print(file1.readlines())**

**print()**

**file1.close()**

**Output:**

**# Python program to illustrate Append vs write mode**

**file1 = open("myfile.txt","w")**

**L = ["This is Delhi \n","This is Paris \n","This is London \n"]**

**file1.writelines(L)**

**file1.close()**

**# Append-adds at last**

**file1 = open("myfile.txt","a")#append mode**

**file1.write("Today \n")**

**file1.close()**

**file1 = open("myfile.txt","r")**

**print("Output of Readlines after appending")**

**print(file1.readlines())**

**print()**

**file1.close()**

**# Write-Overwrites**

**file1 = open("myfile.txt","w")#write mode**

**file1.write("Tomorrow \n")**

**file1.close()**

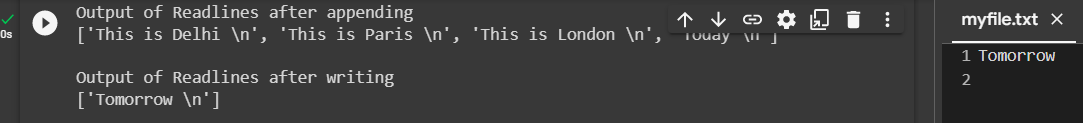
**file1 = open("myfile.txt","r")**

**print("Output of Readlines after writing")**

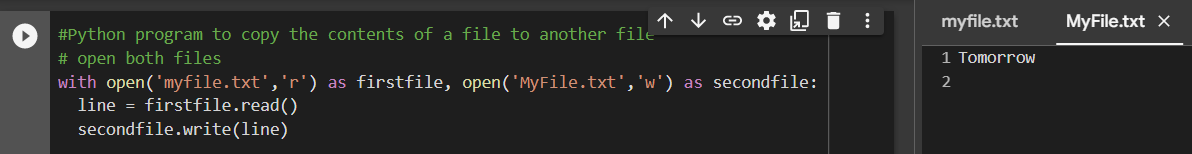
**print(file1.readlines())**

**print()**

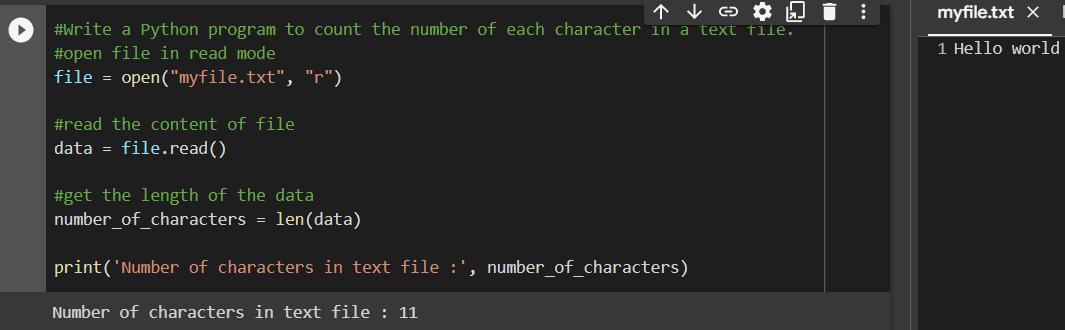
**file1.close()**

**Output:**

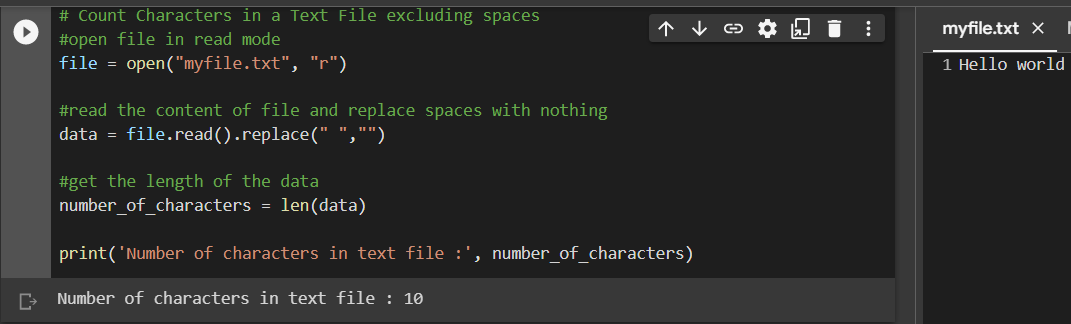
**Program to copy the contents of one file into another**

****

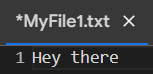
**Program to count the number of each character in a text file**

****

**Program to count the number of each character in a text file excluding spaces**

****

**Python program to print each line of a file in reverse order**

****

**#Python program to print each line of a file in reverse order**

**# Open the file in write mode**

**f1 = open("MyFile1.txt", "w")**

**with open("MyFile1.txt", "r") as myfile:**

**data = myfile.read()**

**# For Full Reversing we will store the value of data into new variable data\_1 in a reverse order using [start: end: step],**

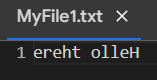
**#where step when passed -1 will reverse the string**

**data\_1 = data[::-1]**

**# Now we will write the fully reverse data in the output1 file using following command**

**f1.write(data\_1)**

**#f1.close()**

****

**Conclusion:**

In this experiment, we studied and implemented the concept of file handling in python.

I faced some difficulties such as improper indentation in functions.

**Shashwat Tripathi**

**D10A 58**

**BATCH: C**

**PYTHON LAB**

**Assignment No: 7**

Aim: Write python programs to understand -

Designing Graphical user interface (GUI) using built-in tools in python (Tkinter, PyQt, Kivy etc.).

Code:

import tkinter as tk

import customtkinter as ctk

ctk.set\_appearance\_mode("dark")

ctk.set\_default\_color\_theme("green")

# create the window

root = tk.Tk()

root.title("Classroom Attendance System")

root.geometry('800x500')

root.configure(bg='black')

# ctk.set\_appearance\_mode("dark")

ctk.set\_default\_color\_theme("green")

scrollable\_frame = ctk.CTkScrollableFrame(master=root, width=200, height=200, )

scrollable\_frame.pack(pady=20, padx=60, fill="both", expand=True)

# create the labels

name\_label = ctk.CTkLabel(master=scrollable\_frame, text="Name", font=("Comic Sans MS",10,"bold"))

name\_label.grid(row=0, column=0)

name\_label.pack(pady=(12, 0), padx=10)

name\_entry = ctk.CTkEntry(master=scrollable\_frame)

# name\_entry.grid(row=0, column=1)

name\_entry.pack()

date\_label = ctk.CTkLabel(master=scrollable\_frame, text="Date", font=("Comic Sans MS",10,"bold"))

# date\_label.grid(row=1, column=0)

date\_label.pack(pady=(12, 0), padx=10)

date\_entry = ctk.CTkEntry(master=scrollable\_frame)

# date\_entry.grid(row=1, column=1)

date\_entry.pack()

subject\_label = ctk.CTkLabel(master=scrollable\_frame, text="Subject", font=("Comic Sans MS",10,"bold"))

# subject\_label.grid(row=2, column=0)

subject\_label.pack(pady=(12, 0), padx=10)

subject\_entry = ctk.CTkEntry(master=scrollable\_frame)

# subject\_entry.grid(row=2, column=1)

subject\_entry.pack()

time\_label = ctk.CTkLabel(master=scrollable\_frame, text="Time", font=("Comic Sans MS",10,"bold"))

# time\_label.grid(row=3, column=0)

time\_label.pack(pady=(12, 0), padx=10)

time\_entry = ctk.CTkEntry(master=scrollable\_frame)

# time\_entry.grid(row=3, column=1)

time\_entry.pack()

# create the entry boxes

def submit\_attendance():

# get the input from the entry boxes

name\_value = name\_entry.get()

date\_value = date\_entry.get()

subject\_value = subject\_entry.get()

time\_value = time\_entry.get()

textbox1 = ctk.CTkTextbox(root)

textbox1.insert("0.0", f"{name\_value} attended {subject\_value} class on {date\_value} at {time\_value}\n") # insert at line 0 character 0

textbox1.configure(state="disabled", bg\_color='black')

textbox2 = ctk.CTkTextbox(root)

textbox2.insert("0.0", f"{name\_value}") # insert at line 0 character 0

textbox2.configure(state="disabled", bg\_color='black') # configure textbox to be read-only

# create the button

submit\_button = ctk.CTkButton(master=scrollable\_frame, text="SUBMIT", command=submit\_attendance, font=("Comic Sans MS",10,"bold"))

# submit\_button.grid(row=4, column=0)

submit\_button.configure(root)

submit\_button.pack(pady=(12, 0), padx=10)

# create the display box

display\_box = tk.Text(master=scrollable\_frame, height=5, width=40)

# display\_box.grid(row=5, column=1)

display\_box.pack(pady=12, padx=10)

# create the listbox

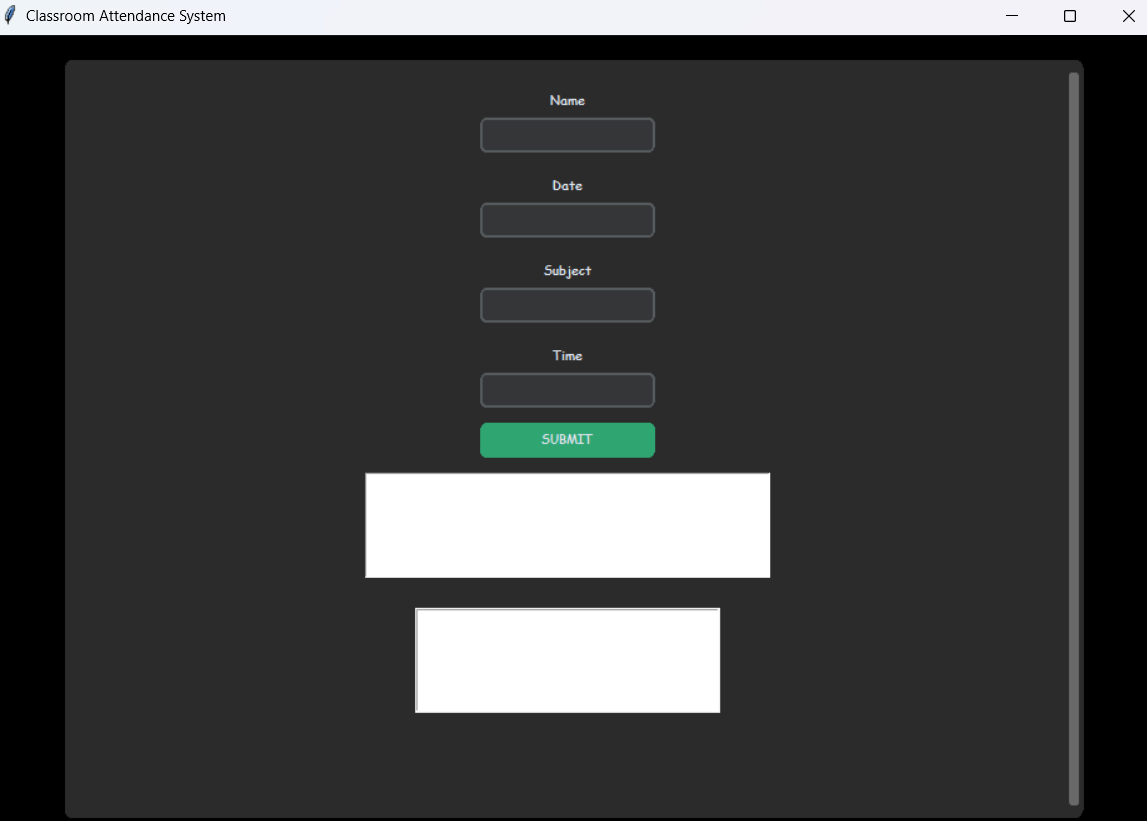
listbox = tk.Listbox(master=scrollable\_frame, height=5, width=40)

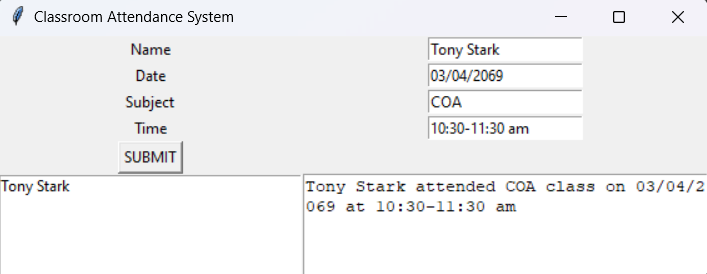
# listbox.grid(row=5, column=0)

listbox.pack(pady=12, padx=10)

# run the window

root.mainloop()





**Conclusion:** In the above experiment, we implemented inbuilt GUI tools in python. I created a Graphical User Interface for attendance management system using tkinter.

I faced difficulties in placing the components at the desired place.

**Shashwat Tripathi**

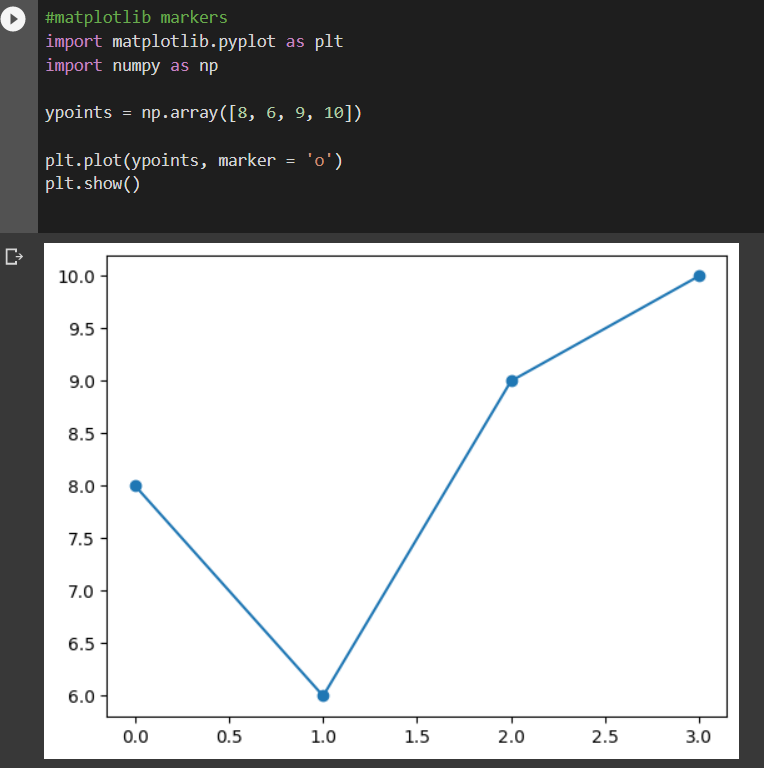
**D10A 58**

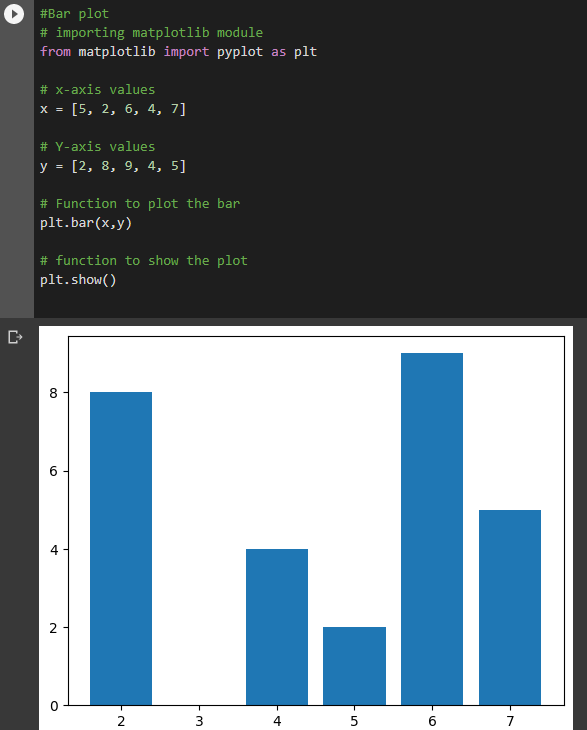
**BATCH: C**

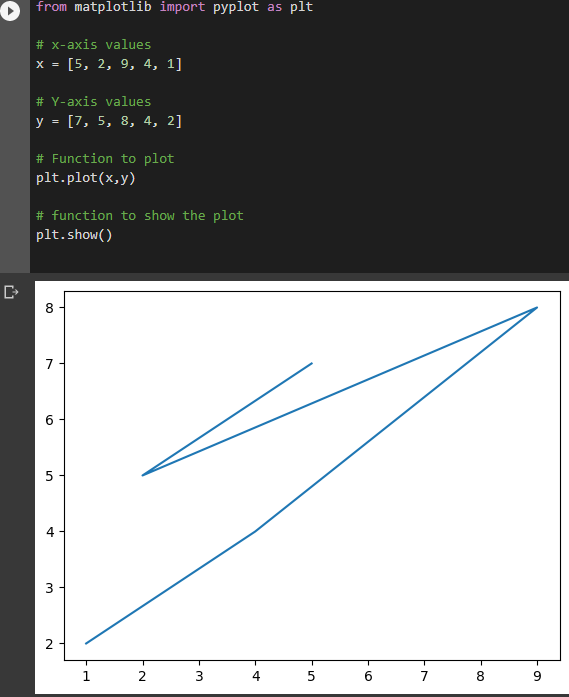
**PYTHON LAB**

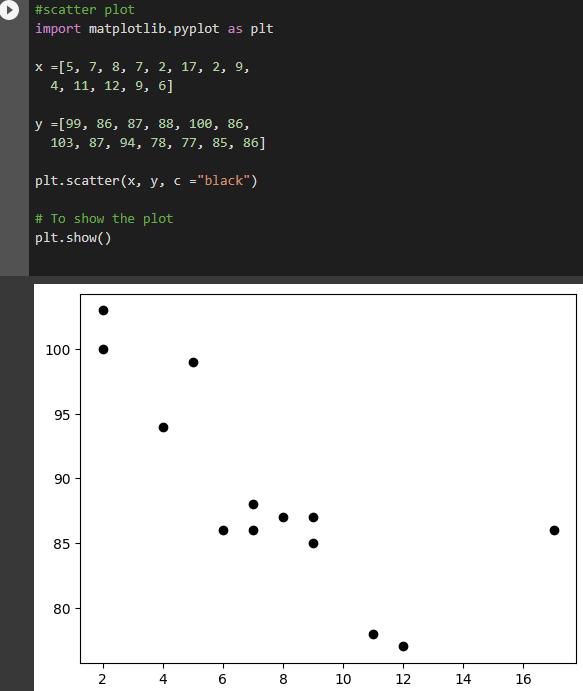
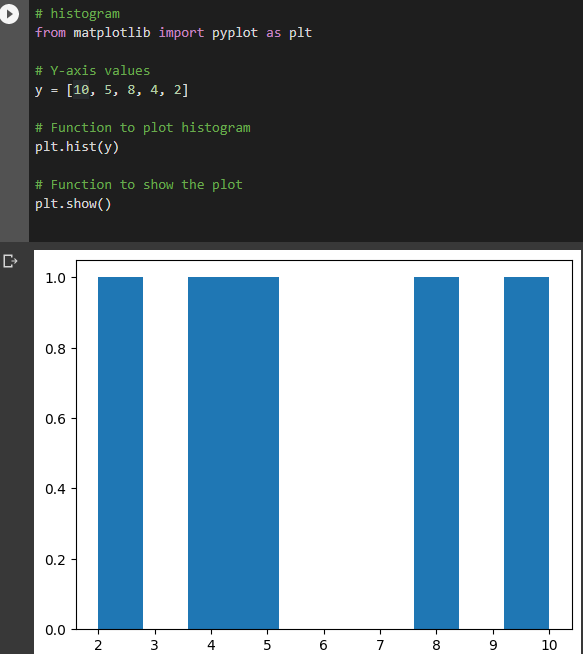
**Assignment No: 8**

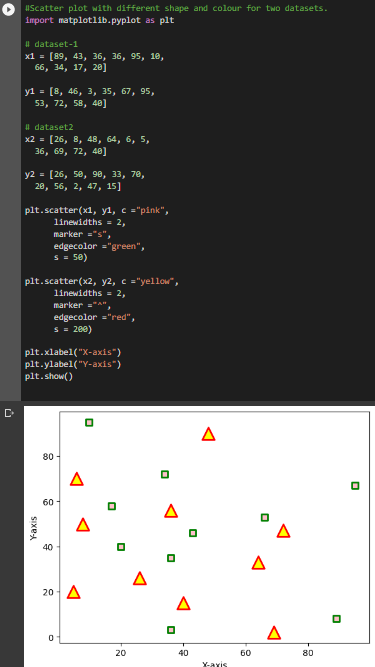
Aim: Write python programs to implement different types of plots using Numpy and Matplotlob.



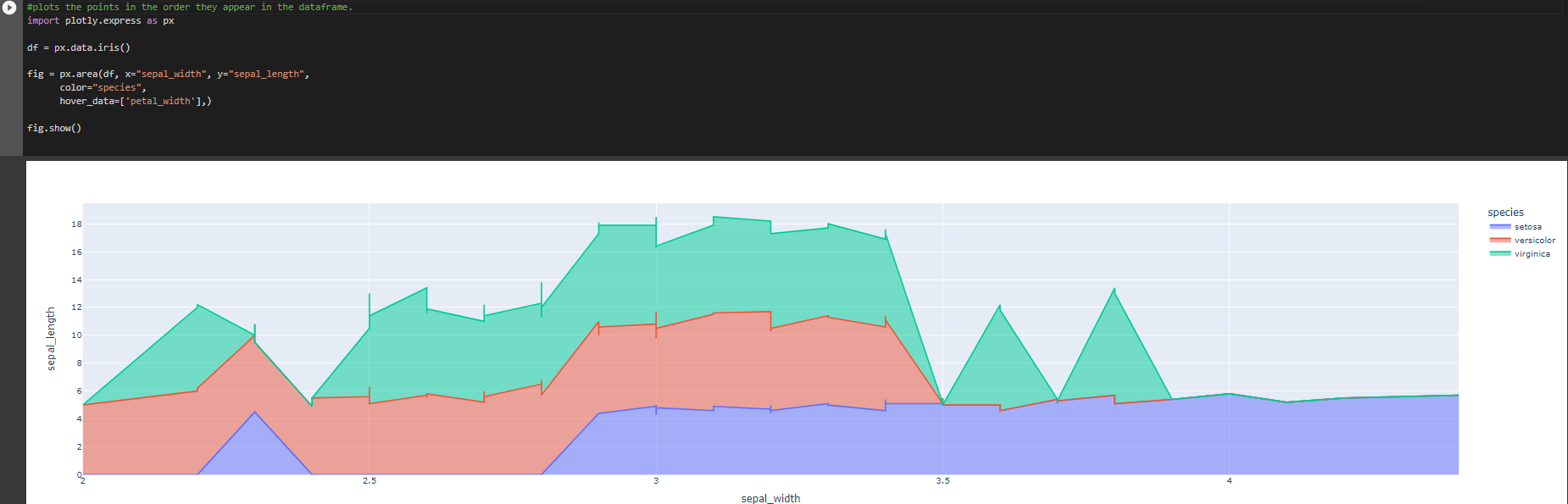


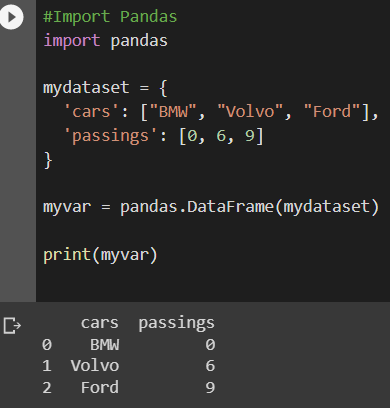


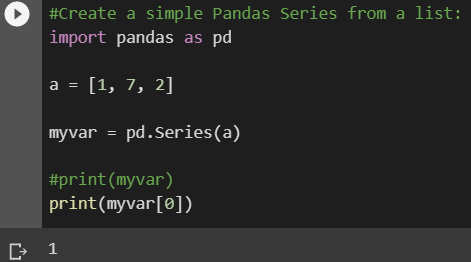


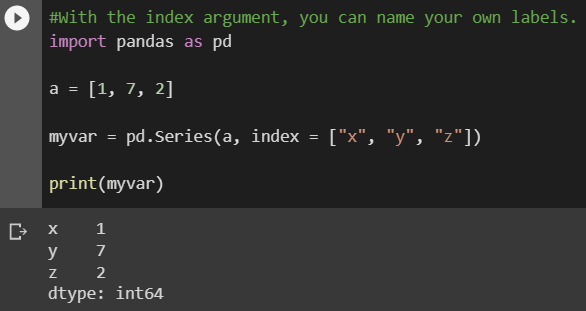


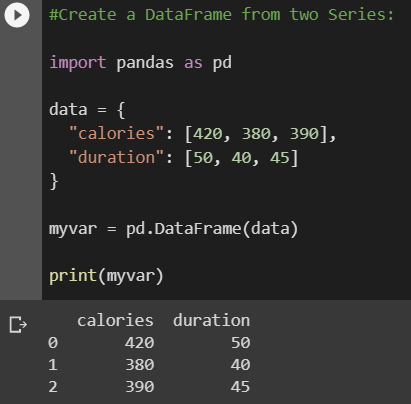












# concatenation

import pandas as pd

# Define a dictionary containing employee data

data1 = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj'],

'Age':[27, 24, 22, 32],

'Address':['Nagpur', 'Kanpur', 'Allahabad', 'Kannuaj'],

'Qualification':['Msc', 'MA', 'MCA', 'Phd']}

# Define a dictionary containing employee data

data2 = {'Name':['Abhi', 'Ayushi', 'Dhiraj', 'Hitesh'],

'Age':[17, 14, 12, 52],

'Address':['Nagpur', 'Kanpur', 'Allahabad', 'Kannuaj'],

'Qualification':['Btech', 'B.A', 'Bcom', 'B.hons']}

# Convert the dictionary into DataFrame

df = pd.DataFrame(data1,index=[0, 1, 2, 3])

# Convert the dictionary into DataFrame

df1 = pd.DataFrame(data2, index=[4, 5, 6, 7])

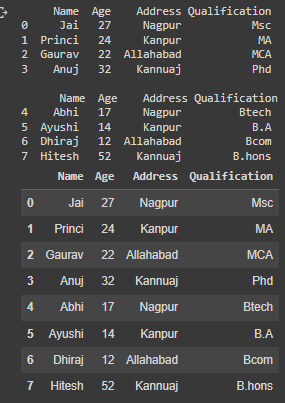
print(df, "\n\n", df1)

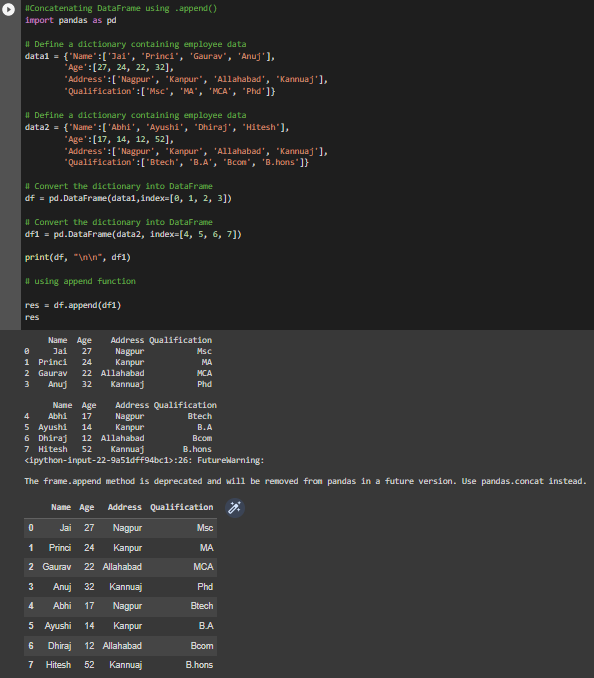
# using a .concat() method

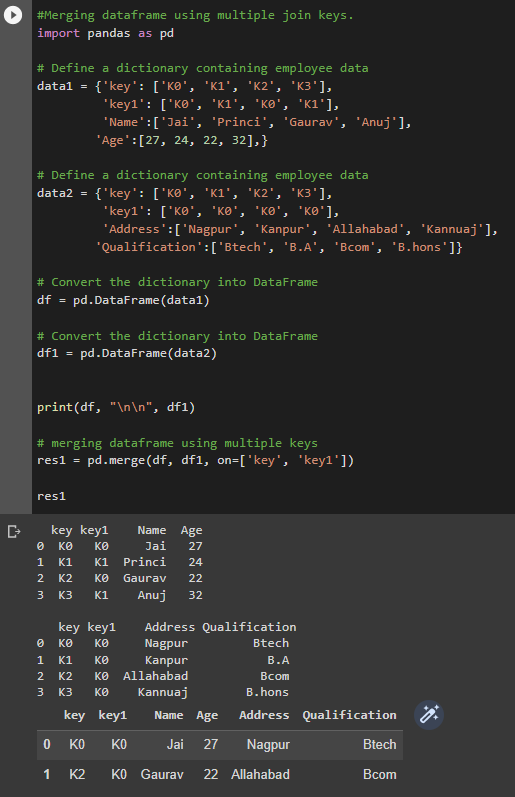
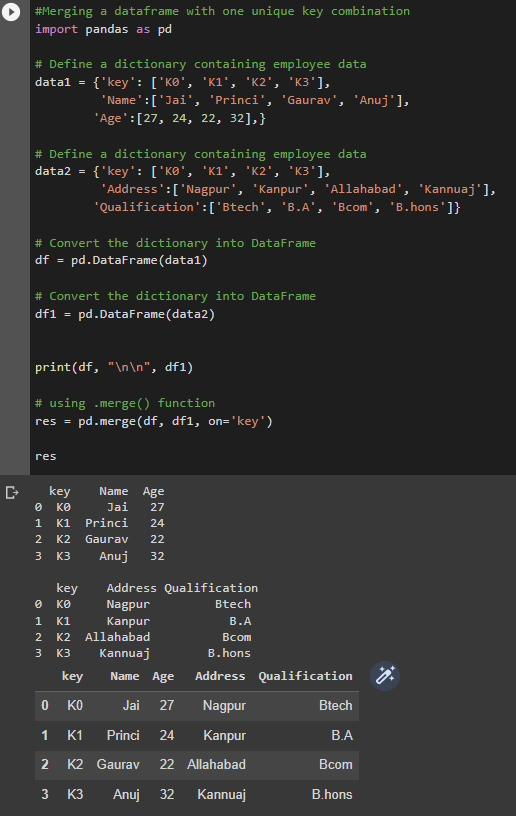
frames = [df, df1]

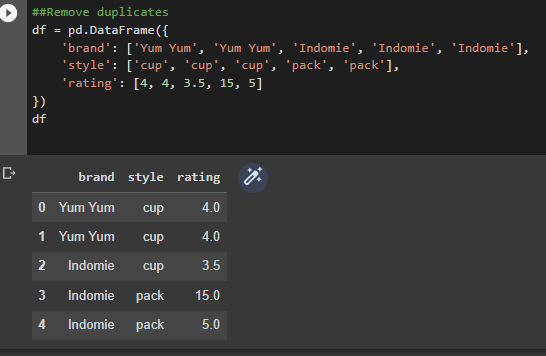
res1 = pd.concat(frames)

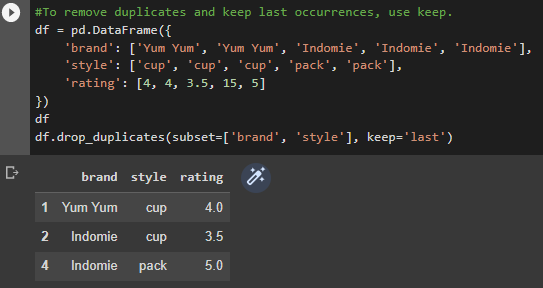
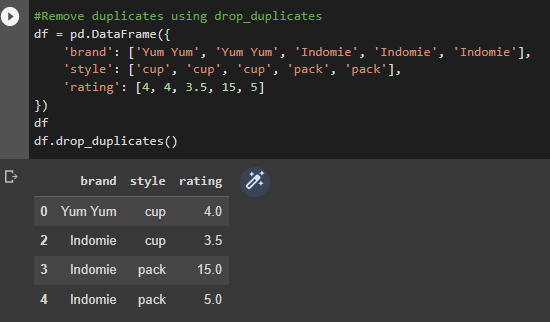
res1











**Conclusion:** In this experiment, we studied and understood about matplotlib library that is used to plot graphs in python.

I faced some syntax errors while creating data frames.

**Shashwat Tripathi**

**D10A 58**

**BATCH: C**

**PYTHON LAB**

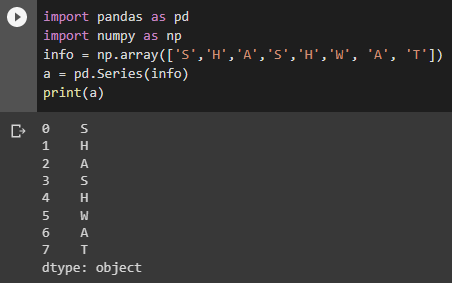
**Assignment No: 9**

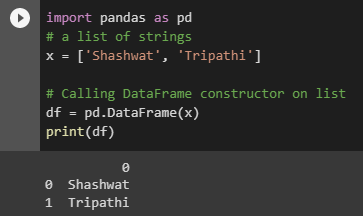
Aim:

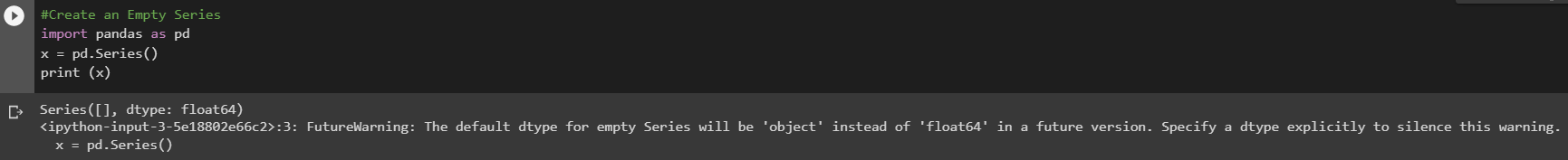
a) Basic operations using pandas like series, data frames, indexing, filtering, combining and merging data frames.

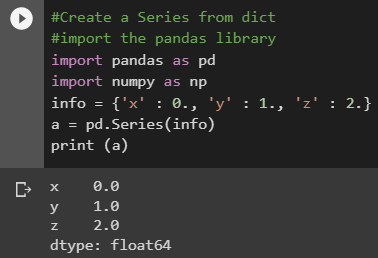
b) Different Linear algebra functions using Scipy.

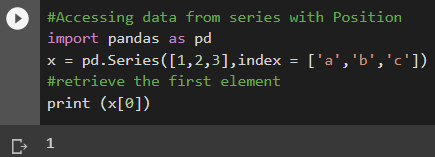
1. Pandas:

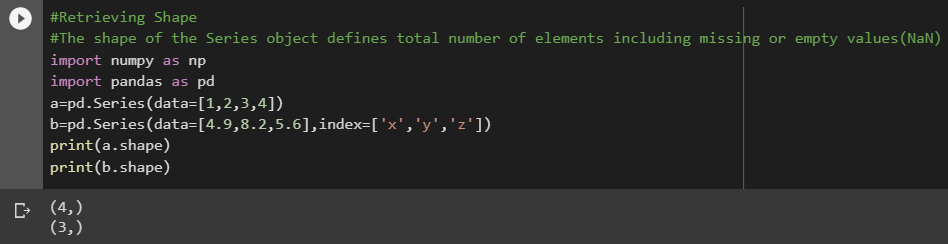


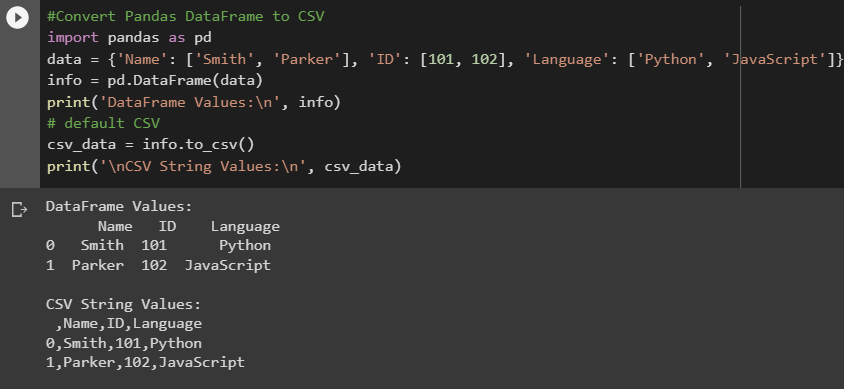




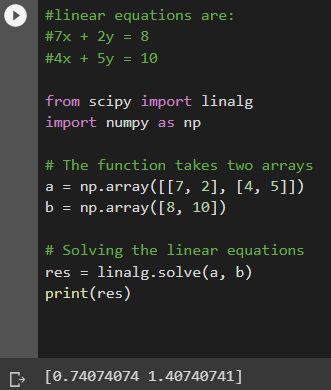


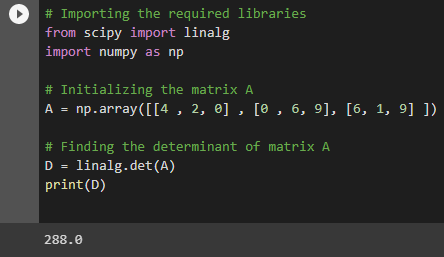


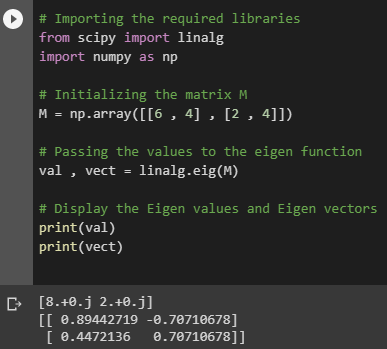




1. Scipy







**Conclusion:**By performing this experiment we understood the basic operations like series, data frames, indexing etc in Pandas and Linear algebra functions in Scipy.

I faced some errors in using numpy and scipy but I resolved them.

**Shashwat Tripathi**

**D10A 58**

**BATCH: C**

**PYTHON LAB**

**Assignment No: 10**

**Create a basic Flask application.**

**Code:**

from flask import Flask, render\_template\_string, request

app = Flask(\_\_name\_\_)

# HTML template with JavaScript code for the whiteboard

whiteboard\_template = '''

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Whiteboard</title>

<script>

var canvas, ctx, color, thickness, is\_drawing;

function start(event) {

is\_drawing = true;

ctx.beginPath();

ctx.moveTo(event.clientX - canvas.offsetLeft, event.clientY - canvas.offsetTop);

}

function draw(event) {

if (is\_drawing) {

ctx.lineTo(event.clientX - canvas.offsetLeft, event.clientY - canvas.offsetTop);

ctx.strokeStyle = color;

ctx.lineWidth = thickness;

ctx.stroke();

}

}

function stop(event) {

is\_drawing = false;

}

function clear\_canvas() {

ctx.clearRect(0, 0, canvas.width, canvas.height);

}

window.onload = function() {

canvas = document.getElementById("whiteboard");

ctx = canvas.getContext("2d");

color = "#000000";

thickness = 1;

is\_drawing = false;

canvas.addEventListener("mousedown", start);

canvas.addEventListener("mousemove", draw);

canvas.addEventListener("mouseup", stop);

canvas.addEventListener("mouseout", stop);

document.getElementById("color-picker").onchange = function() {

color = this.value;

}

document.getElementById("thickness-slider").onchange = function() {

thickness = this.value;

}

document.getElementById("clear-button").onclick = function() {

clear\_canvas();

}

}

</script>

</head>

<body>

<canvas id="whiteboard" width="800" height="600"></canvas>

<br>

<input type="color" id="color-picker">

<input type="range" id="thickness-slider" min="1" max="20" value="1">

<button id="clear-button">Clear</button>

</body>

</html>

'''

@app.route('/')

def whiteboard():

return render\_template\_string(whiteboard\_template)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Output:**

