Name: Manav Shah
Roll No:DSE Student
Second Year CS
Subject: **Programming Lab1**

Experiment No. 1

AIM:

1. Write a program to find square root of a number
2. Write a program to find the Fibonacci Series
3. Write a program to calculate sum of all elements of a array using recursion
4. Write a program to find prime numbers

DESCRIPTION:

1. In Square root program, we first take the input number from user(n). We then iterate from i=1 to n/2+1 to and check if i x i = n. Then i is the square root of the given number. We printed the square root. If not found, we displayed the message that n is not square.

2. In Fibonacci series the first two terms are 0 and 1. All other terms are obtained by adding the preceding two terms. This means to say the nth term is the sum of (n-1)th and (n-2)th term. Here we use the condition statements and for loop to find the number.

3. Recursion is a method of programming or coding a problem, in which a function calls itself one or more times in its body. Usually, it is returning the return value of this function call. A recursive function terminates, if with every recursive call the solution of the problem is downsized and moves towards a base case. A base case is a case, where the problem can be solved without further recursion. A recursion can end up in an infinite loop, if the base case is not met in the call. Thus, to find the sum of elements of an array, we define a recursive function.

4. In prime number program, we take the input number from the user. Then I pass it to a function isPrime(n) .we will declare a counter c = 0. We will iterate a loop from i=2 to n-1 and if n%i == 0 then increment the counter c=c+1. If at the end of loop if c = 0 it is prime number else it is not a prime number.

PROGRAM:

 # 1. Write a program to find square root of a number

```
[23] print ("Enter the number for finding square root : ")
     n=int(input())
     if n==1:
           print("Square root of ",n," is ",1)
     else:
           res = False
           for i in range(1,n//2):
               if i*i == n:
                   res = True
                   print("Square root of ",n," is ",i)
                   break
           if res == False:
               print("No square root for number",n)
```

OUTPUT:

```
Enter the number for finding square root :
81
Square root of  81  is  9
```

 # 2. Write a program to find the Fibonacci Series

```
[12]
     n=int(input("Enter the number of elements in fibonacci series "))

     if(n==1):
         print(0)
     else:
         print(0)
         a=0
         b=1
         c= a+b
         print(c)
         for i in range(2,n):
             c = a+b
             print(c)
             a=b
             b=c
```

OUTPUT:

```
Enter the number of elements in fibonacci series 10
0
1
1
2
3
5
8
13
21
34
```

 # 3. Write a program to calculate sum of all elements of a array using recursion

```
[13] summ =0
     def addele(a,index):
         global summ
         if index == len(a):
             return
         summ = summ + a[index]
         addele(a,index+1)

     a=[]
     n=int(input("ENTER THE NUMBER OF ELEMENTS IN LIST : "))
     for i in range(0,n):
         x=int(input("Enter element:"))
         a.append(x)
     addele(a,0)
     print("SUM OF ALL ELEMENTS : ", summ)
```

OUTPUT:

```
ENTER THE NUMBER OF ELEMENTS IN LIST : 5
Enter element:10
Enter element:20
Enter element:30
Enter element:40
Enter element:50
SUM OF ALL ELEMENTS :  150
```

 # 4. Implement service to find prime numbers

```
[30] def isPrime(n):
         c=0
         for i in range(2,n):
             if n % i == 0:
                 c = c + 1
         if c == 0:
             return True
         else:
             return False


     n=int(input("Enter the number to check for prime : "))
     res = isPrime(n)
     if res == True:
       print(n," is a prime number")
     else:
       print(n,"is not a prime number")
```

OUTPUT:

```
Enter the number to check for prime : 37
37  is a prime number
```

CONCLUSION:

Thus, from this experiment I implemented the basic programs of Python Programming Language using math library for calculating the square root of a given number, if else for finding the Fibonacci series for a number of terms, looping and conditional statements to find the prime numbers using Sieves method and function recursion for calculating the sum of elements in an array .

Name: Manav Shah
Roll No:DSE Student
Second Year CS
Subject: **Programming Lab1**

Experiment No. 2

AIM: Write a program using list comprehension

DESCRIPTION:

List comprehension is an elegant way to define and create list in python. We can create lists just like mathematical statements and in one line only. List comprehensions provide us with a simple way to create a list based on some iterable. During the creation, elements from the iterable can be conditionally included in the new list and transformed as needed.
A list comprehension generally consists of these parts:

- Output expression,
- input sequence,
- a variable representing member of input sequence and
- an optional predicate parts.

For example:
lst = [x ** 2 for x in range (1, 11) if x % 2 == 1]
here: -

- x ** 2 is output expression,
- range (1, 11) is input sequence,
- x is variable and
- if x % 2 == 1 is predicate part.

PROGRAM:

 # 1. Write a program to find the square of the number

```
l = [1,2,3,4,5]
l2 = [i ** 2 for i in l]
print(l2)
```

OUTPUT:

```
[1, 4, 9, 16, 25]
```

 # 2. Write a program to find vowels in string

```
s=input("Enter a string : ")
l2 = [i for i in s if (i == 'a' or i == 'e' or i == 'i' or i == 'o' or i == 'u' or i == 'A' or i == 'E' or i == 'I' or i == 'O' or i == 'U')]
print(l2)
```

OUTPUT:

```
Enter a string : UMBrella
['U', 'e', 'a']
```

 # 3. Write a program to find length of all words in the list less than 5 characters

```
l = ["helloo","world","umbrella","off","runn","swim"]
l1 = [ i for i in l if len(i)< 5]
print(l1)
```

OUTPUT:

```
['off', 'runn', 'swim']
```

CONCLUSION:

Thus, from this experiment I implemented the programs in Python Programming Language using list comprehension that allows to reduce the lines of code. I executed programs for find vowels , find words whose length is less than 5 and square of numbers in list.

Name: Manav Shah
Roll No:DSE Student
Second Year CS
Subject: **Programming Lab1**

Experiment No. 3

AIM: Write programs to analyze lists using map, reduce and filter methods.

THEORY :

● **LIST** : A Python list is an ordered and changeable collection of data objects. Unlike an array, which can contain objects of a single type, a list can contain a mixture of objects. SYNTAX : List_name=[contents]

● **LAMBDA FUNCTIONS :** Python Lambda Functions are anonymous functions meaning that the function is without a name. This function can have any number of arguments but only one expression, which is evaluated and returned. The lambda keyword is used to define an anonymous function in Python. SYNTAX : lambda arguments : expression ➢ Map, Filter, and Reduce are paradigms of functional programming. They allow the programmer (you) to write simpler, shorter code, without necessarily needing to bother about intricacies like loops and branching.

● **MAP** : The map() function in Python takes in a function and a list as an argument. The function is called with a lambda function and a list and a new list is returned which contains all the lambda-modified items returned by that function for each item. It can take any number of iterables. SYNTAX : map(funct, iterables)

● **FILTER** : The filter() function in Python takes in a function and a list as arguments.While map() passes each element in the iterable through a function and returns the result of all elements having passed through the function, filter(), first of all, requires the function to return boolean values (true or false) and then passes each element in the iterable through the function, "filtering" away those that are false. It takes only one iterable. SYNTAX : filter(func, iterable) ● **REDUCE** : The reduce() function in Python takes in a function and a list as an argument. The function is called with a lambda function and an iterable and a new reduced result is returned. This performs a repetitive operation over the pairs of the iterable. reduce() applies a function of two arguments cumulatively to

the elements of an iterable, optionally starting with an initial argument. The reduce() function belongs to the functools module. SYNTAX : reduce(func, iterable)

CODE 1

Python Program to find square of each element in list using map and lambda

```
[8]  n=int(input("How many numbers you want to add in list : "))
     print("Enter ",n," elements")
     l=[]
     for i in range(0,n):
       l.append(int(input()))
     square = list(map(lambda x: x**2,l))
     print(square)
```

```
How many numbers you want to add in list : 5
Enter  5  elements
12
13
11
19
23
[144, 169, 121, 361, 529]
```

CODE 2

Python program to filter element in list as even and odd using filter and lambda

```
n=int(input("How many numbers you want to add in list : "))
print("Enter ",n," elements")

l=[]

for i in range(0,n):
  l.append(int(input()))

odd = filter(lambda x: x % 2 != 0, l)
print("Odd elements in list : ",list(odd))

even = filter(lambda x: x % 2 == 0, l)
print("Even elements in list : ",list(even))
```

```
How many numbers you want to add in list : 7
Enter  7  elements
23
56
77
20
11
56
87
Odd elements in list :  [23, 77, 11, 87]
Even elements in list :  [56, 20, 56]
```

CODE 3

Python program to add all elements in list using reduce and lambda

```
from functools import reduce
n=int(input("How many numbers you want to add in list : "))
print("Enter ",n," elements")

l=[]

for i in range(0,n):
  l.append(int(input()))

print("Result of adding all numbers in list : ")
addList = reduce(lambda x,y: x + y,l)
print(addList)
```

```
How many numbers you want to add in list : 5
Enter  5  elements
34
45
21
67
88
Result of adding all numbers in list :
255
```

CODE 4

Python program to check if the first letter of the name is capital.

```
n=int(input("How many names you want to add in list : "))
print("Enter ",n," names")

l=[]

for i in range(0,n):
  l.append(input())

names = list(filter(lambda a:a[0].isupper(),l))
print("Names with first letter capital ", names)
```

```
How many names you want to add in list : 5
Enter  5  names
Manav
John
paul
Pete
Marsh
Names with first letter capital  ['Manav', 'John', 'Pete', 'Marsh']
```

CONCLUSION : Thus we learnt about 3 different built-in functions of python map() ,reduce() and filter() and understand their uses in different instances. We also learnt to reduce the lines of code and also how to avoid using loops using these functions.Using these we have coded multiple amounts of different codes in which we have implemented these functions.

Name: Manav Shah
Roll No:DSE Student
Second Year CS
Subject: **Programming Lab1**

Experiment No. 4

AIM: Write an contact application using dictionary

THEORY :

Dictionaries are used to store data values in key:value pairs. A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

CODE 1:

To write a contact application using dictionary in python

```
contact_dict = {}
index = 0
def add_contact():
     global index
     name = input("Enter the Name: ")
     if name in contact_dict.values():
       print(f"{name} already exits")
       return
     verify = False
     while not verify:
       email = input("Enter the Email: ")
       if "@" not in email:
           print("email must include @")
           print("****Email must be in this format : example121@gmail.com
****")
           print("")
       if "." not in email:
```

```
            print("email must include .")
            print("****Email must be in this format : example121@gmail.com
****")
            print("")
        else:
            verify = True
    verify = False
    while not verify:
        phone = input("Enter the Mobile Number: ")
        if len(phone)!=10 or phone.isdigit() == False:
            print("Enter valid phone number")
            print("****Phone Number must be of 10 digits and must not contain
other characters****")
            print("")
        else:
            verify =True
    contact_dict[index] = {"Name":name,"Email": email, "Phone": phone}
    index = index +1
    print("Contact added successfully\n")

def edit_contact():
        idx = 0
        name = input("Enter the Name whose details are to be updated: ")
        for key,val in contact_dict.items():
            if name not in val["Name"]:
                print(f"No contact named {name} found\n")
            else:
                idx = key
                email = input("Enter the Updated Email: ")
                phone = input("Enter the Updated Mobile Number: ")
                contact_dict[idx]["Email"] = email
                contact_dict[idx]["Phone"] = phone
                print("Contact updated Successfully\n")
```

```
def delete_contact():
     idx =0
     name = input("Enter the Name whose details are to be Deleted: ")
     for key,val in contact_dict.copy().items():ma
         if name not in val["Name"]:
           print(f"No contact named {name} found\n")
         else:
           idx = key
           contact_dict.pop(idx)
           print("Contact deleted Successfully\n")


def display():
 if len(contact_dict) == 0:
     print(f"No contacts found\n")
 else:
    print("Available Details")
    for value in contact_dict.values():
       print(f"Name : {value['Name']}")
       print(f"Mail Id : {value['Email']}")
       print(f"Phone Number : {value['Phone']}")
       print()

while True:
 print("1. Add New Contact")
 print("2. Edit a Contact")
 print("3. Delete a Contact")
 print("4. Display Contacts")
 print("5:Exit")
 choice = int(input("Enter your choice: "))
```

```
match choice:
  case 1:
    add_contact()

  case 2:
    edit_contact()

  case 3:
    delete_contact()
  case 4:
    display()
  case 5:
    break
```

Output:

```
1. Add New Contact
2. Edit a Contact
3. Delete a Contact
4. Display Contacts
5:Exit
Enter your choice: 1
Enter the Name: manav
Enter the Email: mshah@gmail.com
Enter the Mobile Number: 1234567890
Contact added successfully

1. Add New Contact
2. Edit a Contact
3. Delete a Contact
4. Display Contacts
5:Exit
Enter your choice: 4
Available Details
Name : manav
Mail Id : mshah@gmail.com
Phone Number : 1234567890

1. Add New Contact
2. Edit a Contact
3. Delete a Contact
4. Display Contacts
5:Exit
Enter your choice: 2
Enter the Name whose details are to be updated: manav
Enter the Updated Email: hello@world.com
Enter the Updated Mobile Number: 9876543210
Contact updated Successfully
```

```
Enter the Updated Email: [illegible]
Enter the Updated Mobile Number: 9876543210
Contact updated Successfully

1. Add New Contact
2. Edit a Contact
3. Delete a Contact
4. Display Contacts
5:Exit
Enter your choice: 3
Enter the Name whose details are to be Deleted: manav
Contact deleted Successfully

1. Add New Contact
2. Edit a Contact
3. Delete a Contact
4. Display Contacts
5:Exit
Enter your choice: 5
```

CONCLUSION : In this experiment, we learnt about dictionaries in python. We created a contact application in python using dictionaries wherein a user can enter contact details,edit them as well as update them. We also provided functions to display all the contact details.We also provided validations to mobile number and email id.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 5

AIM: To write an application to edit a text file.

THEORY :

File handling in Python is a powerful and versatile tool that can be used to perform a wide range of operations. However, it is important to carefully consider the advantages and disadvantages of file handling when writing Python programs, to ensure that the code is secure, reliable, and performs well.

Python treats files differently as text or binary and this is important. Each line of code includes a sequence of characters and they form a text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. Let's start with the reading and writing files.

CODE:

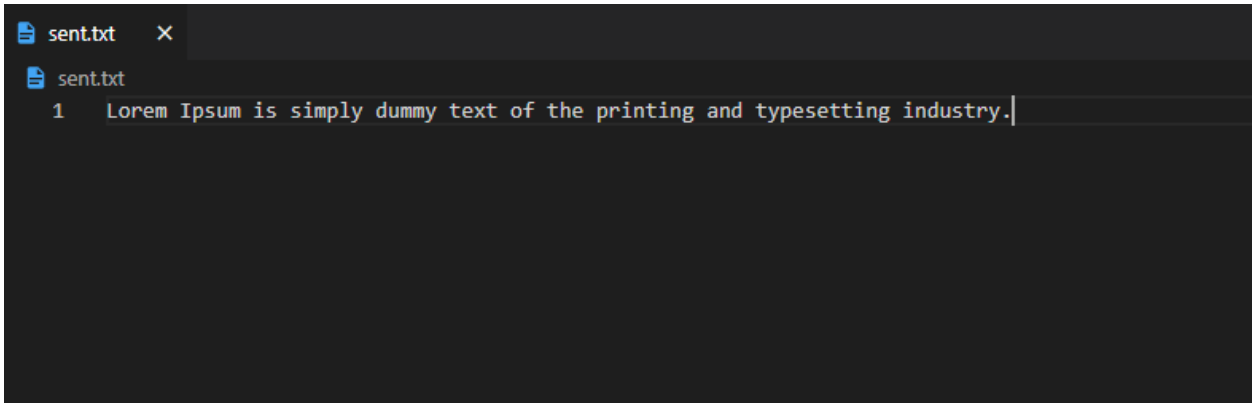
To write an application to edit a text file.

```
def read_file_contents(file_name="sent.txt"):
    file = open(file_name, "r")
    print(f"Text in the file is: {file.read()}\n")
    file.close()

read_file_contents()
text = input("Enter the text to be added to the file: ")
file = open("sent.txt", "a")
```

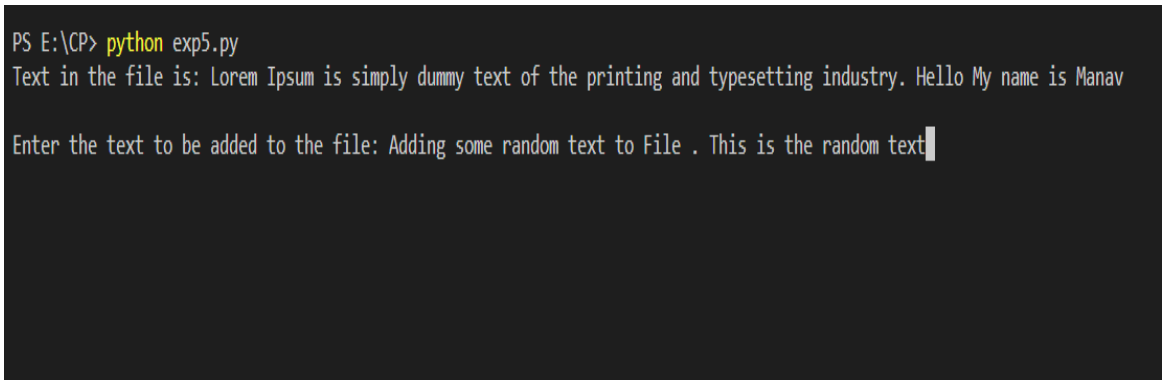
```
file.write(" " + text)

file.close()


read_file_contents()
```

OUTPUT:

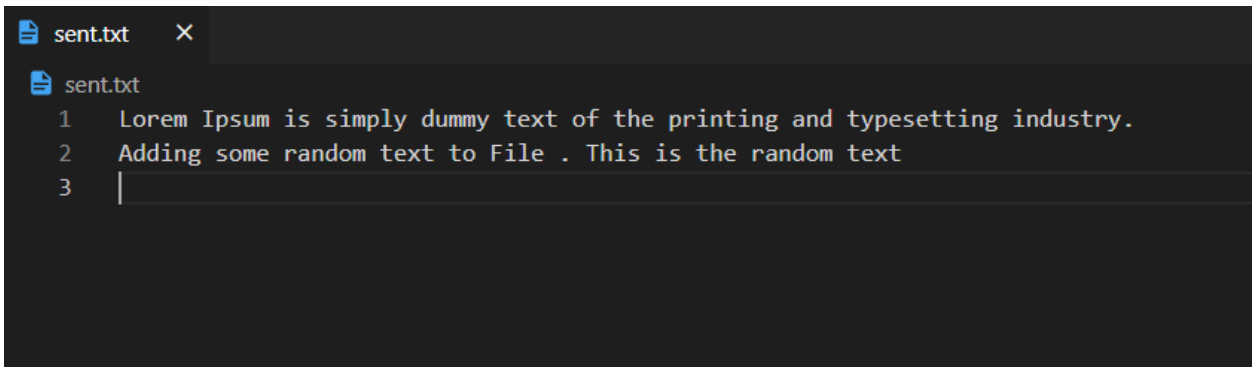
Contents of the file before Editing:



Program Output



Contents of the file after Editing:



CONCLUSION: From this experiment, we learnt about File handling in python. We created an application to edit a text file and add more content to the text file by taking input from the user. We can perform file handling in python by opening the file using the open() function provided by python.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 6

AIM: To create a calculator in python using Tkinter

THEORY :
Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

CODE 1:
To create a calculator in python using Tkinter

```
from tkinter import *


win = Tk()

win.title("Tkinter Calculator")

win.resizable(False, False)


def button_press(num):

global exp

exp= input_text.get()

exp= str(exp) + str(num)

input_text.set(exp)
```

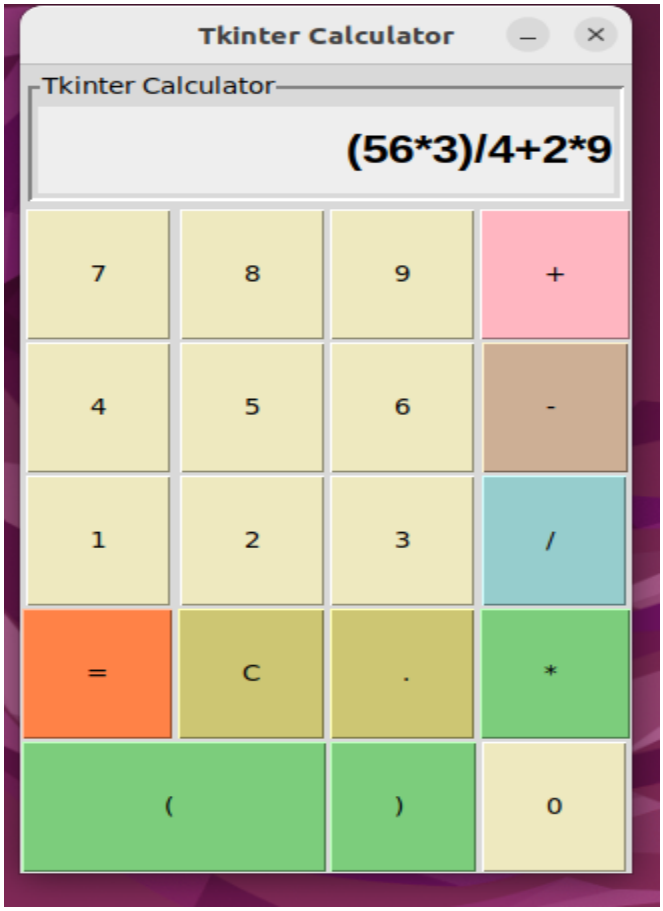
```
def button_oper(opr):
global exp
exp = input_text.get()
exp= str(exp) + str(opr)
input_text.set(exp)
global oper
oper = opr



def clr_scr():
global exp
input_text.set("")
exp = ""

def equal(event):
try:
global display
global oper
global exp
exp = input_text.get()
input_text.set("")
result = float(eval(exp))
exp = result
display.insert(0, result)
input_text.set(result)
except Exception as e:
input_text.set(e)




display_frame = LabelFrame(win, text="Tkinter Calculator", relief=SUNKEN,
```

```
padx=2, pady=2)
display_frame.grid(row=0, column=0, columnspan=4, padx=2, pady=4)


input_text = StringVar()


display = Entry(display_frame, font=('arial', 18, 'bold'),
textvariable=input_text, width=22, bg="#eee", bd=0, justify=RIGHT)
display.pack(ipady= 12)




button_1 = Button(win, padx=30, pady=30, text="1", bg="LemonChiffon2",
command=lambda: button_press(1))
button_2 = Button(win, padx=30, pady=30, text="2", bg="LemonChiffon2",
command=lambda: button_press(2))
button_3 = Button(win, padx=30, pady=30, text="3", bg="LemonChiffon2",
command=lambda: button_press(3))
button_4 = Button(win, padx=30, pady=30, text="4", bg="LemonChiffon2",
command=lambda: button_press(4))
button_5 = Button(win, padx=30, pady=30, text="5", bg="LemonChiffon2",
command=lambda: button_press(5))
button_6 = Button(win, padx=30, pady=30, text="6", bg="LemonChiffon2",
command=lambda: button_press(6))
button_7 = Button(win, padx=30, pady=30, text="7", bg="LemonChiffon2",
command=lambda: button_press(7))
button_8 = Button(win, padx=30, pady=30, text="8", bg="LemonChiffon2",
command=lambda: button_press(8))
button_9 = Button(win, padx=30, pady=30, text="9", bg="LemonChiffon2",
command=lambda: button_press(9))
button_0 = Button(win, padx=30, pady=30, text="0", bg="LemonChiffon2",
command=lambda: button_press(0))
button_add = Button(win, padx=30, pady=30, text="+", bg="light pink",
command=lambda: button_oper("+"))
button_sub = Button(win, padx=30, pady=30, text="- ", bg="Peachpuff3",
```

```
command=lambda: button_oper("-"))
button_div = Button(win, padx=30, pady=30, text="/ ", bg="PaleTurquoise3",
command=lambda: button_oper("/"))
button_mul = Button(win, padx=30, pady=30, text="* ", bg="paleGreen3",
command=lambda: button_oper("*"))
button_or = Button(win, padx=30, pady=30,width=11, text="( ",
bg="paleGreen3", command=lambda: button_oper("("))
button_cr = Button(win, padx=30, pady=30, text=") ", bg="paleGreen3",
command=lambda: button_oper(")"))
button_equal = Button(win, padx=30, pady=30, text="=", bg="sienna1",
command=lambda: equal(""))
button_clear = Button(win, padx=30, pady=30, text="C", bg="khaki3",
command=clr_scr)
button_dec = Button(win, padx=30, pady=30, text=" .", bg="khaki3",
command=lambda: button_press("."))

button_7.grid(row=1, column=0)
button_8.grid(row=1, column=1)
button_9.grid(row=1, column=2)
button_add.grid(row=1, column=3)

button_4.grid(row=2, column=0)
button_5.grid(row=2, column=1)
button_6.grid(row=2, column=2)
button_sub.grid(row=2, column=3)

button_1.grid(row=3, column=0)
button_2.grid(row=3, column=1)
button_3.grid(row=3, column=2)
button_div.grid(row=3, column=3)

button_equal.grid(row=4, column=0)
button_clear.grid(row=4, column=1)
button_dec.grid(row=4, column=2)
```

```
button_mul.grid(row=4, column=3)


button_or.grid(row=5,column=0,columnspan=2)
button_cr.grid(row=5,column=2)
button_0.grid(row=5,column=3)



win.bind("<Return>", equal)



win.mainloop()
```

OUTPUT



CONCLUSION : In this experiment, we learnt about to create GUI in python using Tkinter. We created a calculator in python wherein a user can enter any arithmetic expression ,edit them as well as update them and he will get the desired result of the expression in output.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 7

AIM: To write a code in which a client sends a file to a server and server saves the file and prints the file contents.

THEORY :

Socket Programming: Python socket programming allows communication between devices over a network using sockets, facilitating data exchange through client-server architecture. It's a versatile tool for building applications like chat programs and networked games.

The socket module in Python may be used to create an echo server and client. When a message arrives, the server code waits for incoming connections and echoes the message to the client. A connection is established with the server, a message is sent, and the client code shows the echoed answer.

CODE:

To write a code in which a client sends a file to a server and server saves the file and prints the file contents.

client.py

```
import os
import socket


client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(("localhost", 9999))
file = open("sent.txt", "rb")
file_size = os.path.getsize("sent.txt")
```

```
client.send("received.txt".encode())
client.send(str(file_size).encode())

data = file.read()
client.sendall(data)
client.send(b"<END>")

print("File sent successfully")

file.close()
client.close()
```

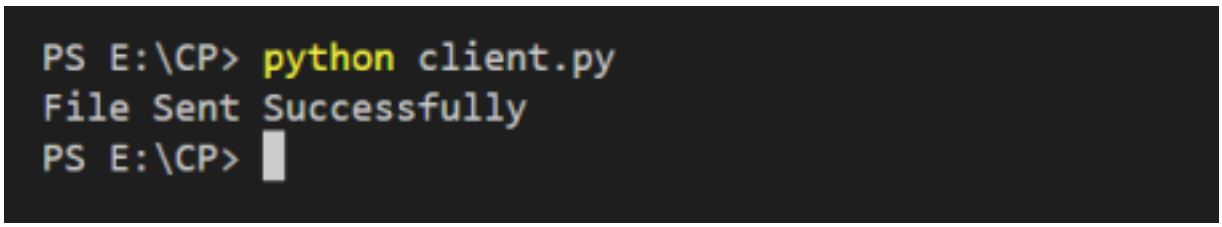
server.py

```
import socket

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(("localhost", 9999))

server.listen()

client, addr = server.accept()

file_size = client.recv(1024).decode()

file = open("recieved.txt", "wb")
file_bytes = b""

done = False
while not done:
    data = client.recv(1024)
    if(file_bytes[-5:] == b"<END>"):
```

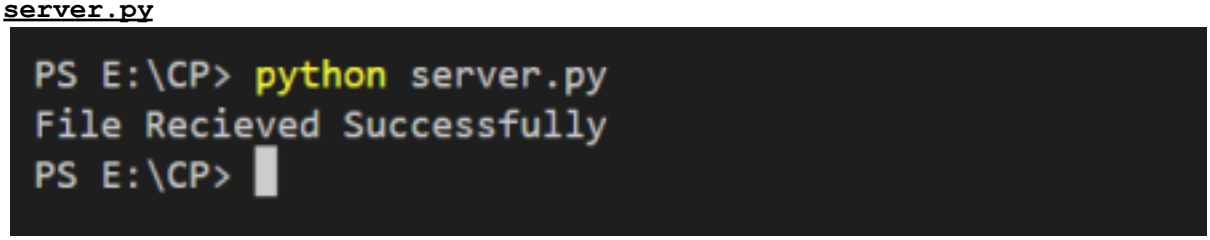
```
            done = True
        else:
            file_bytes += data

    file_bytes = file_bytes[:-5]
    file_bytes = file_bytes[2:]
    file.write(file_bytes)

    print("File Recieved Successfully")

    file.close()
    server.close()
    client.close()
```

OUTPUT:

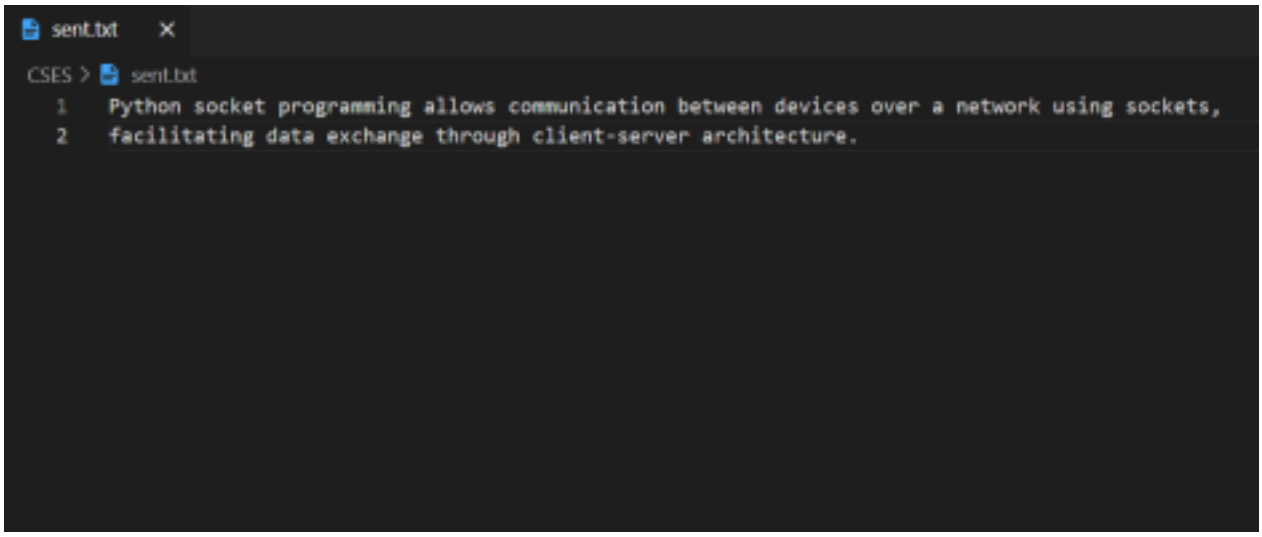
client.py



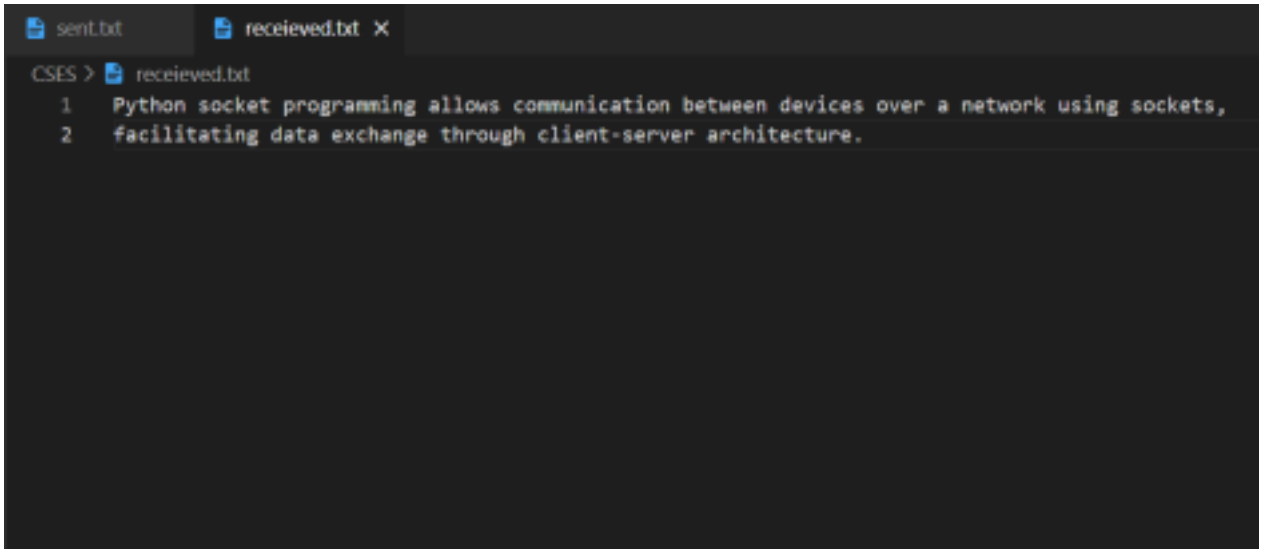
server.py



Sent File Contents:



Received File Contents:



CONCLUSION : From this experiment, we learnt about Socket programming in python, with the help of which we can communicate with different nodes in a network using their IP addresses. In this experiment, we created a client and a server, then connected the client with the server over the network, then we shared a file from the client to the server over the network and the server copies and prints the contents of the file.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 8

AIM: Write a program in a python to perform CRUD operations on database(sqlite3/MySql)

THEORY :

Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases.

SQLite is a lightweight, serverless, self-contained, and open-source relational database management system. In Python, you can interact with SQLite using the sqlite3 **module.**

CRUD stands for Create, Read, Update, and Delete—a set of basic operations used in database management systems to manipulate data.

CODE:

```
[2] import sqlite3
```

```
conn = sqlite3.connect("Student.db")
```

```
[12] conn.execute('''
        CREATE TABLE Departments(
          Code INTEGER KEY NOT NULL,
          Branch VARCHAR NOT NULL,
          StudentCount INTEGER NOT NULL
        );
    ''')
    conn.commit()
    print("Department Table Created")
```

```
Department Table Created
```

```
conn.execute('''
    CREATE TABLE STUDENTS(
      Branch INTEGER NOT NULL,
      ID INTEGER NOT NULL,
      Name VARCHAR NOT NULL,
      LastName VARCHAR NOT NULL,
      CONSTRAINT fk_Departments_Code FOREIGN KEY(Branch) REFERENCES DEPARTMENT(C
    );
''')
conn.commit()
print("Students table created")
```

```
Students table created
```

```
[14] conn.execute('''INSERT INTO Departments VALUES(01,'Computer Science',10)''')
     conn.execute('''INSERT INTO Departments VALUES(02,'IT',20)''')
     conn.execute('''INSERT INTO Departments VALUES(03,'EXTC',30)''')
     conn.execute('''INSERT INTO Departments VALUES(04,'Mech',40)''')
     conn.commit()
```

```
conn.execute('''INSERT INTO STUDENTS VALUES(01,101,'Manav','Shah')''')
conn.execute('''INSERT INTO STUDENTS VALUES(02,102,'John','Doe')''')
conn.execute('''INSERT INTO STUDENTS VALUES(02,103,'Pop','Alae')''')
conn.execute('''INSERT INTO STUDENTS VALUES(01,104,'DBdk','Shh')''')
conn.execute('''INSERT INTO STUDENTS VALUES(04,105,'iwef','ewfbhi')''')
conn.execute('''INSERT INTO STUDENTS VALUES(03,106,'wdn','wbhfe')''')
conn.execute('''INSERT INTO STUDENTS VALUES(02,107,'man','Shah')''')
conn.execute('''INSERT INTO STUDENTS VALUES(03,108,'Jonny','Doe')''')
conn.execute('''INSERT INTO STUDENTS VALUES(04,109,'Paul','Alae')''')
conn.execute('''INSERT INTO STUDENTS VALUES(03,110,'ewf','ue')''')
conn.commit()
```

```
[24] data = conn.execute(f'''SELECT * FROM Departments;''')
     for row in data:
         print(row)
```

```
(1, 'Computer Science', 10)
(2, 'IT', 20)
(3, 'EXTC', 30)
(4, 'Mech', 40)
```

```
data = conn.execute(f'''SELECT * FROM STUDENTS''')
for row in data:
    print(row)
```

```
(1, 101, 'Manav', 'Shah')
(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(1, 104, 'DBdk', 'Shh')
(4, 105, 'iwef', 'ewfbhi')
(3, 106, 'wdn', 'wbhfe')
(2, 107, 'man', 'Shah')
(3, 108, 'Jonny', 'Doe')
(4, 109, 'Paul', 'Alae')
(3, 110, 'ewf', 'ue')
```

```
[28] n = input("Enter the Branch Code to be Searched (01 : CS,02 : IT , 03 : EXTC, 04 : Mech);")

     data = conn.execute(f'''
         SELECT * FROM STUDENTS
         WHERE Branch IN({n});
     ''')

     for row in data:
         print(row)
```

```
Enter the Branch Code to be Searched (01 : CS,02 : IT , 03 : EXTC, 04 : Mech);2
(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(2, 107, 'man', 'Shah')
```

```python
print("Students data before updating : \n")
data = conn.execute(f'''
    SELECT * FROM STUDENTS
''')

for row in data:
    print(row)
print('\n')
conn.execute('''UPDATE STUDENTS SET ID = 100 WHERE Name = 'Manav' ''')
print("Students data after updating : \n")
data = conn.execute(f'''
    SELECT * FROM STUDENTS
''')

for row in data:
    print(row)
print('\n')
```

```
Students data before updating :

(1, 101, 'Manav', 'Shah')
(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(1, 104, 'DBdk', 'Shh')
(4, 105, 'iwef', 'ewfbhi')
(3, 106, 'wdn', 'wbhfe')
(2, 107, 'man', 'Shah')
(3, 108, 'Jonny', 'Doe')
(4, 109, 'Paul', 'Alae')
(3, 110, 'ewf', 'ue')


Students data after updating :

(1, 100, 'Manav', 'Shah')
(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(1, 104, 'DBdk', 'Shh')
(4, 105, 'iwef', 'ewfbhi')
(3, 106, 'wdn', 'wbhfe')
(2, 107, 'man', 'Shah')
(3, 108, 'Jonny', 'Doe')
(4, 109, 'Paul', 'Alae')
(3, 110, 'ewf', 'ue')
```

```python
print("Students data before deleting : \n")
data = conn.execute(f'''
    SELECT * FROM STUDENTS
''')

for row in data:
    print(row)
print('\n')
conn.execute('''DELETE FROM STUDENTS WHERE Name = 'Manav' ''')
print("Students data after updating : \n")
data = conn.execute(f'''
    SELECT * FROM STUDENTS
''')

for row in data:
    print(row)
print('\n')
```

```
Students data before deleting :

(1, 100, 'Manav', 'Shah')
(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(1, 104, 'DBdk', 'Shh')
(4, 105, 'iwef', 'ewfbhi')
(3, 106, 'wdn', 'wbhfe')
(2, 107, 'man', 'Shah')
(3, 108, 'Jonny', 'Doe')
(4, 109, 'Paul', 'Alae')
(3, 110, 'ewf', 'ue')


Students data after updating :

(2, 102, 'John', 'Doe')
(2, 103, 'Pop', 'Alae')
(1, 104, 'DBdk', 'Shh')
(4, 105, 'iwef', 'ewfbhi')
(3, 106, 'wdn', 'wbhfe')
(2, 107, 'man', 'Shah')
(3, 108, 'Jonny', 'Doe')
(4, 109, 'Paul', 'Alae')
(3, 110, 'ewf', 'ue')
```

CONCLUSION : In this experiment, we performed a program in a python to perform CRUD operations on a database (sqlite3). We created a table named Departments to store the information related to different departments such as branch, dept code and and other table name Students to store the information related to different students such as name, ID and branch code.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 9

AIM: To plot different graphs using matplotlib library.

THEORY:

Matplotlib:

Matplotlib is a Python library for creating static, animated, and interactive visualizations in a variety of formats. It's a powerful tool for data visualization and is widely used in fields like data science, machine learning, and scientific research.

With Matplotlib, you can create line plots, scatter plots, bar plots, histograms, pie charts, and more. It provides a high level of customization, allowing you to fine-tune your plots to suit your needs.

Matplotlib is a low level graph plotting library in python that serves as a visualization utility. Matplotlib was created by John D. Hunter. Matplotlib is open source and we can use it freely. Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility.

Most of the Matplotlib utilities lie under the **pyplot** submodule, and are usually imported under the **plt** alias.

The **plot()** function is used to draw points (markers) in a diagram. By default, the **plot()** function draws a line from point to point.

Dataset Link:

https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data

CODE:

To plot different graphs using matplotlib library.

```
import pandas as pd
import matplotlib.pyplot as plt
from datasets import load_dataset
import seaborn as sns
from matplotlib.figure import Figure
```

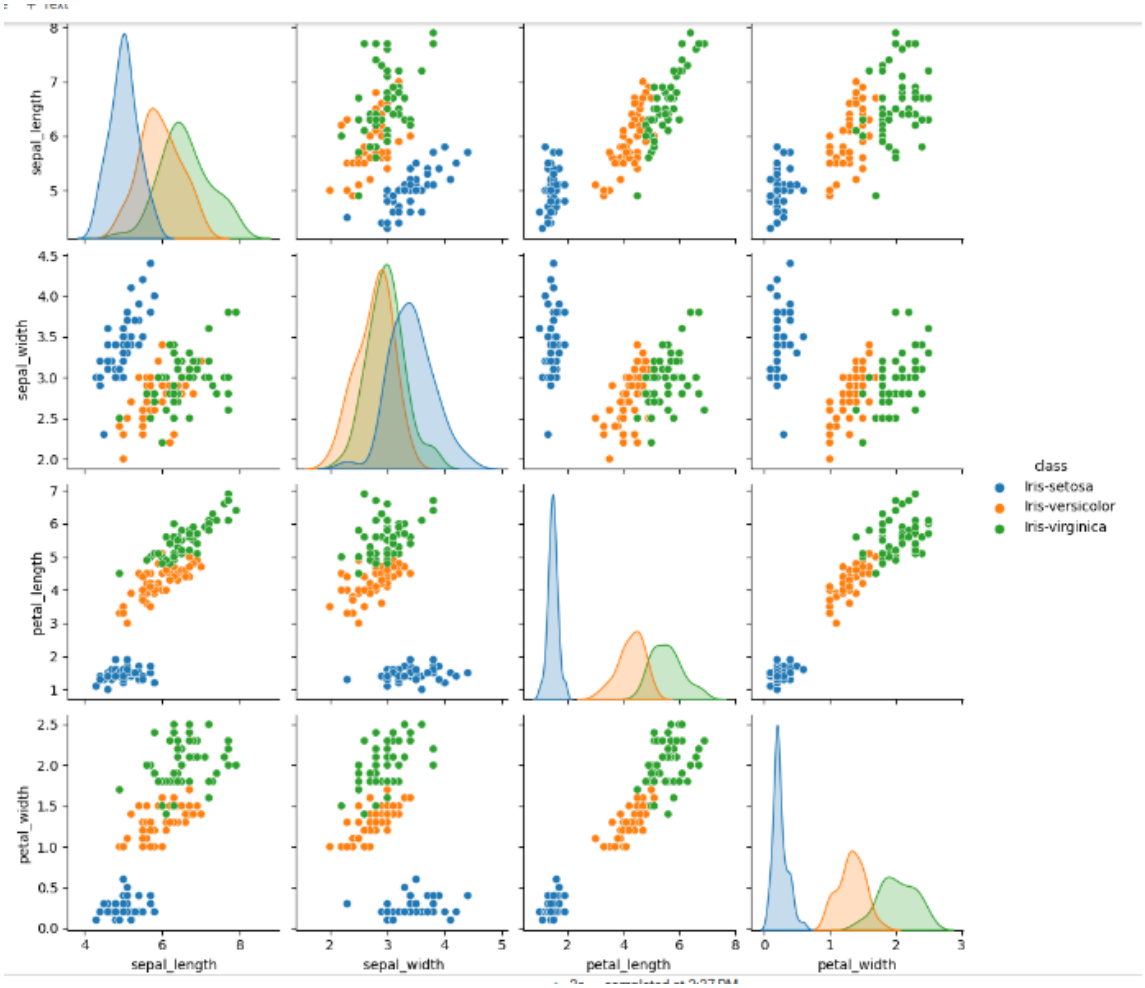
```
cols = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
'class']
df =
pd.read_csv("https://archive.ics.uci.edu/ml/machine-learning-databases/iri
s/iris.data", names=cols)
df.head()
```

Output:

	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
sns.pairplot(df, hue='class');
```

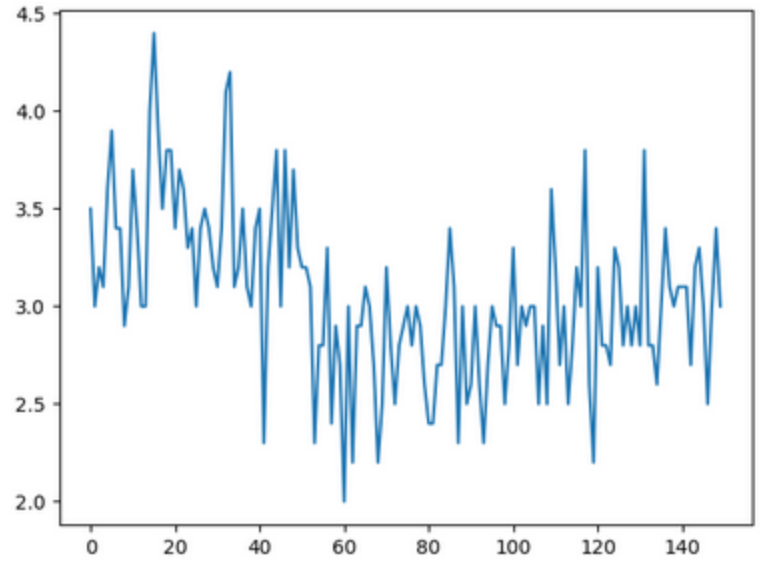
Output:




```
plt.plot(df['sepal_width'])
```

Output:

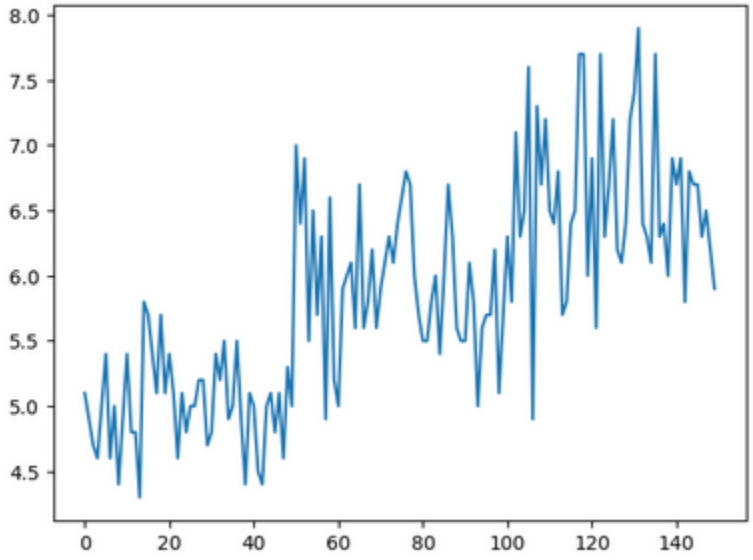
```
[<matplotlib.lines.Line2D at 0x7e5dd28d0370>]
```




```
plt.plot(df['sepal_length'])
```

Output:

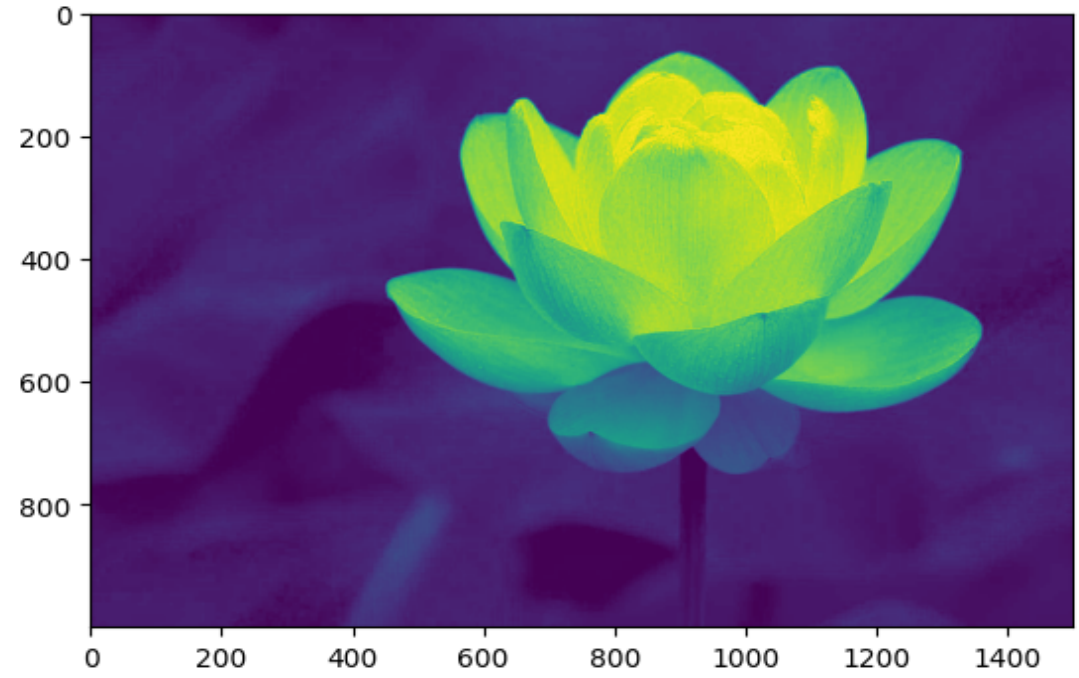
```
[<matplotlib.lines.Line2D at 0x7e5dd28730a0>]
```




```
imgplot = plt.imshow(img)
```

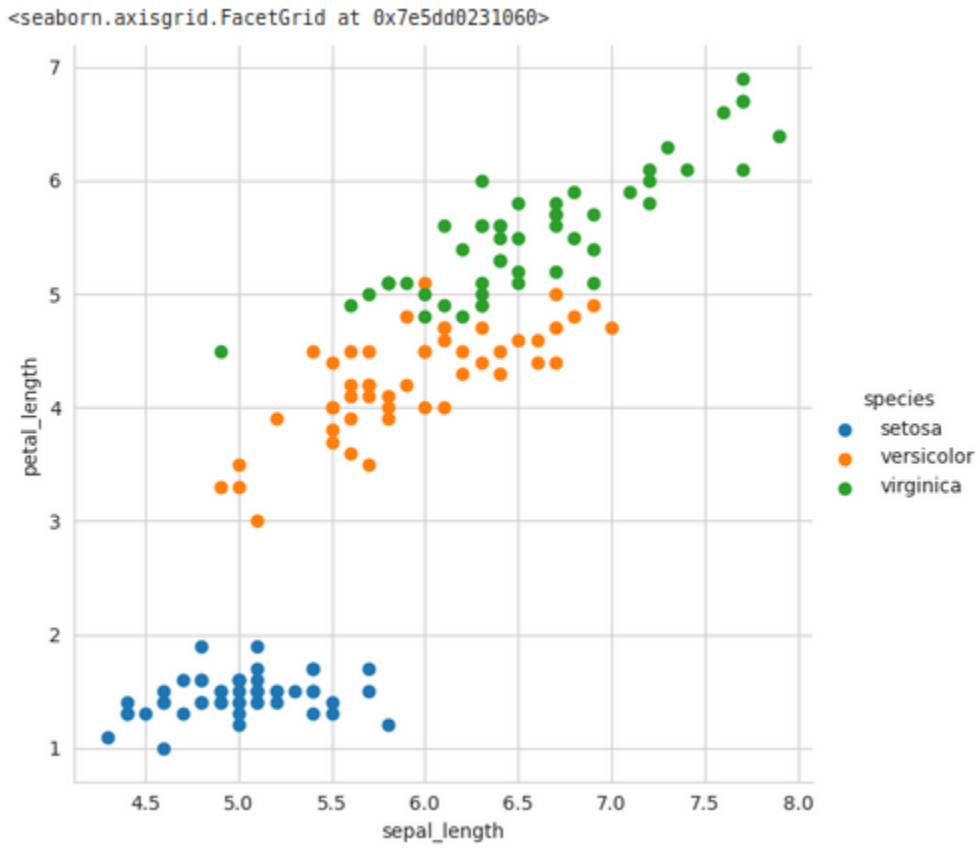

```
lum_img = img[:, :, 0]
plt.imshow(lum_img)
```

```
<matplotlib.image.AxesImage at 0x79f0a5d38cd0>
```



```
iris = sns.load_dataset('iris')
sns.set_style("whitegrid")
sns.FacetGrid(iris, hue ="species",
     height = 6).map(plt.scatter,
             'sepal_length',
             'petal_length').add_legend()
```

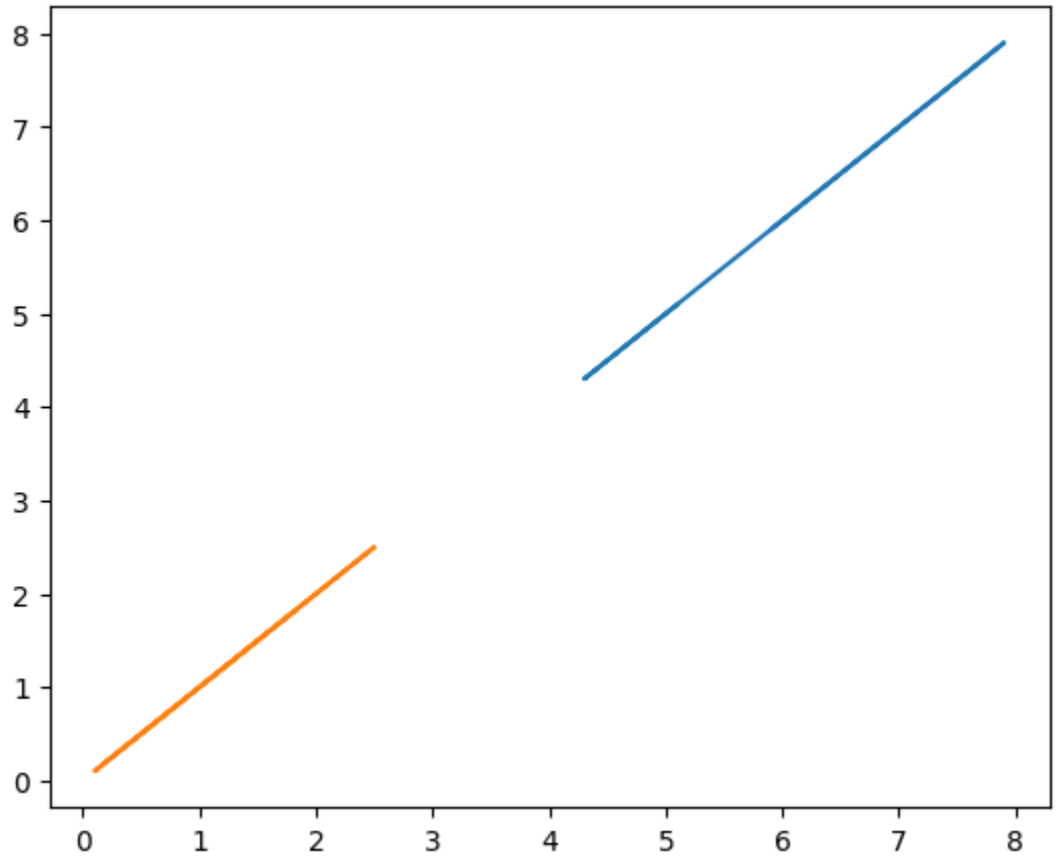
Output:




```
fig = plt.figure(figsize = (5, 4))

# Adding the axes to the figure
ax = fig.add_axes([1, 1, 1, 1])
```

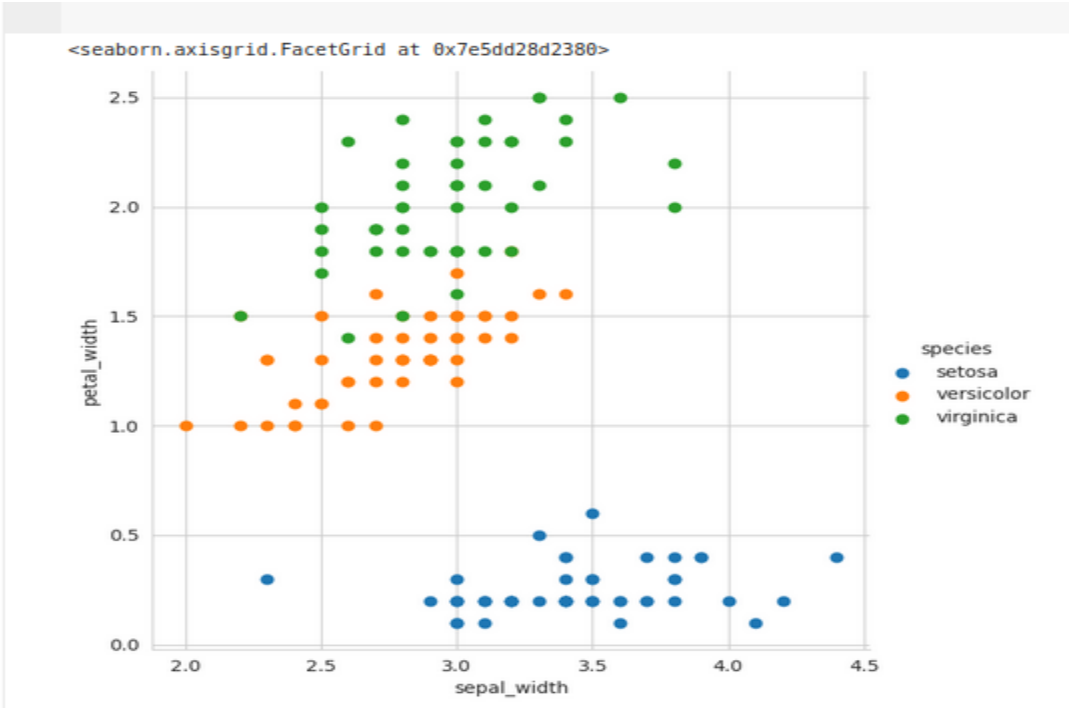
```
# plotting 1st dataset to the figure
ax1 = ax.plot(df["sepal_length"], df["sepal_length"])

# plotting 2nd dataset to the figure
ax2 = ax.plot(df["petal_width"], df["petal_width"])
plt.show()
```

Output:



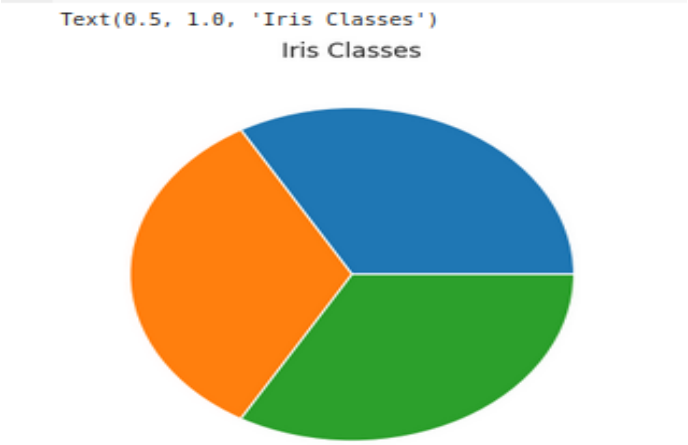

```
iris = sns.load_dataset('iris')
sns.set_style("whitegrid")
sns.FacetGrid(iris, hue ="species",
     height = 6).map(plt.scatter,
             'sepal_width',
             'petal_width').add_legend()
```

Output:




```
setosa_count = sum([i=='Iris-setosa' for i in df['class']])
versicolor_count = sum([i=='Iris-versicolor' for i in df['class']])
virginica_count = sum([i=='Iris-virginica' for i in df['class']])
x = (0.1, 0, 0)
plt.pie([setosa_count, versicolor_count, virginica_count])
plt.title("Iris Classes")
```

Output:



CONCLUSION : From this experiment, we learnt how to plot different graphs using matplotlib library. We worked on the iris dataset and also used the data from the dataset to plot different graphs such as pie chart, bar graph, line chart, scatter plot,working on images by providing color gradients which can be very useful for data analysis and visualization..

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 10

AIM: To perform analysis of data using NumPy and Pandas libraries.

THEORY:

Numpy: NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software.

Features of NumPy:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Pandas: Pandas is a powerful and versatile library that simplifies tasks of data manipulation in Python . Pandas is built on top of the NumPy library and is particularly well-suited for working with tabular data, such as spreadsheets or SQL tables. Its versatility and ease of use make it an essential tool for data analysts, scientists, and engineers working with structured data in Python.

It is built on the top of the NumPy library which means that a lot of structures of NumPy are used or replicated in Pandas. The data produced by Pandas are often used as input for plotting functions of Matplotlib, statistical analysis in SciPy, and machine learning algorithms in Scikit-learn. Here is a list of things that we can do using Pandas.

CODE:

To perform analysis of data using NumPy and Pandas libraries.

```
import pandas as pd
cols = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'class']
df = pd.read_csv("https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data", names=cols)
df.head()
```

Output:

	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
df.to_csv("iris.csv", index=False)
df.head()                    # shows top 5 rows
```

	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
df.tail()                    # shows bottom 5 rows
```

	sepal_length	sepal_width	petal_length	petal_width	class
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

```
df.describe()                    # provides info about the df
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
df.dtypes                        # shows the datatypes of columns
```

```
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
class            object
dtype: object
```

```
new_df = df.copy()               # creates a copy
new_df.to_numpy()
```

```
array([[5.1, 3.5, 1.4, 0.2, 'Iris-setosa'],
       [4.9, 3.0, 1.4, 0.2, 'Iris-setosa'],
       [4.7, 3.2, 1.3, 0.2, 'Iris-setosa'],
       [4.6, 3.1, 1.5, 0.2, 'Iris-setosa'],
       [5.0, 3.6, 1.4, 0.2, 'Iris-setosa'],
       [5.4, 3.9, 1.7, 0.4, 'Iris-setosa'],
       [4.6, 3.4, 1.4, 0.3, 'Iris-setosa'],
       [5.0, 3.4, 1.5, 0.2, 'Iris-setosa'],
       [4.4, 2.9, 1.4, 0.2, 'Iris-setosa'],
       [4.9, 3.1, 1.5, 0.1, 'Iris-setosa'],
       [5.4, 3.7, 1.5, 0.2, 'Iris-setosa'],
       [4.8, 3.4, 1.6, 0.2, 'Iris-setosa'],
       [4.8, 3.0, 1.4, 0.1, 'Iris-setosa'],
       [4.3, 3.0, 1.1, 0.1, 'Iris-setosa'],
       [5.8, 4.0, 1.2, 0.2, 'Iris-setosa'],
       [5.7, 4.4, 1.5, 0.4, 'Iris-setosa'],
       [5.4, 3.9, 1.3, 0.4, 'Iris-setosa'],
       [5.1, 3.5, 1.4, 0.3, 'Iris-setosa'],
       [5.7, 3.8, 1.7, 0.3, 'Iris-setosa'],
```

```
new_df.T                                # Transposes the df
```

	1	2	3	4	5	6	7	8	9	10	...	141	142	143	144	145	146	147
sepal_length	5.1	4.9	4.7	4.6	5.0	5.4	4.6	5.0	4.4	4.9	...	6.7	6.9	5.8	6.8	6.7	6.7	6.3
sepal_width	3.5	3.0	3.2	3.1	3.6	3.9	3.4	3.4	2.9	3.1	...	3.1	3.1	2.7	3.2	3.3	3.0	2.5
petal_length	1.4	1.4	1.3	1.5	1.4	1.7	1.4	1.5	1.4	1.5	...	5.6	5.1	5.1	5.9	5.7	5.2	5.0
petal_width	0.2	0.2	0.2	0.2	0.2	0.4	0.3	0.2	0.2	0.1	...	2.4	2.3	1.9	2.3	2.5	2.3	1.9
class	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	...	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica	Iris-virginica

5 rows × 150 columns

```
new_df.sort_index(axis=0, ascending=False)          # sorts the df
```

	sepal_length	sepal_width	petal_length	petal_width	class
150	5.9	3.0	5.1	1.8	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
146	6.7	3.0	5.2	2.3	Iris-virginica
...	...	...	...	...	...
5	5.0	3.6	1.4	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa

```
new_df.loc[[1, 2], :]                    # shows specified cells
```

	1	2	3	4	5
1	123.0	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa

```
new_df.loc[(new_df[1] < 5) & (new_df[2] > 2)]      # shows specified
cells with condition
```

	1	2	3	4	5
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa
9	4.4	2.9	1.4	0.2	Iris-setosa
10	4.9	3.1	1.5	0.1	Iris-setosa
12	4.8	3.4	1.6	0.2	Iris-setosa
13	4.8	3.0	1.4	0.1	Iris-setosa

```
new_df.drop([5], axis=1, inplace=True)        # used to drop rows and
columns
```

```
new_df
```

	1	2	3	4
1	123.0	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
6	5.4	3.9	1.7	0.4
...	...	...	...	...

```
new_df.drop_duplicates()                    # drops the duplicate records
```

	1	2	3	4
1	123.0	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
6	5.4	3.9	1.7	0.4
...	...	...	...	...

```
new_df.info()                    # shows info about df
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 149 entries, 1 to 150
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   1       149 non-null    float64
 1   2       149 non-null    float64
 2   3       149 non-null    float64
 3   4       149 non-null    float64
dtypes: float64(4)
memory usage: 9.9+ KB
```

```
df.min()                # shows min value from each columns
```

```
sepal_length            4.3
sepal_width             2.0
petal_length            1.0
petal_width             0.1
class           Iris-setosa
dtype: object
```

```
df.max()                # shows maxvalue from each columns
```

```
sepal_length               7.9
sepal_width                4.4
petal_length               6.9
petal_width                2.5
class           Iris-virginica
dtype: object
```

```
df.median()             # shows median value from each columns
```

```
sepal_length    5.80
sepal_width     3.00
petal_length    4.35
petal_width     1.30
dtype: float64
```

```
df.std()                # shows standard deviation value from each columns
```

```
sepal_length    0.828066
sepal_width     0.433594
petal_length    1.764420
petal_width     0.763161
dtype: float64
```

```
df.corr()               # shows correlation between each column with every
other column
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.109369	0.871754	0.817954
sepal_width	-0.109369	1.000000	-0.420516	-0.356544
petal_length	0.871754	-0.420516	1.000000	0.962757
petal_width	0.817954	-0.356544	0.962757	1.000000

CONCLUSION : From this experiment, we learnt about the Numpy and Pandas library in python which are used for data processing and manipulation. Pandas is built on top of Numpy and is well-suited for working with tabular data, such as spreadsheets or SQL tables. In this experiment, we imported the iris dataset and performed various operations on the dataframe like sorting, slicing, transposing, etc. We also cleaned the dataframe by removing the duplicate values.

Name: Manav Shah
Roll No: 231070902
Second Year CS
Subject: **Programming Lab1**

Experiment No. 11

AIM: To perform different image processing operations using the python Pillow library.

THEORY:

Pillow: Python Imaging Library (expansion of PIL) is the de facto image processing package for Python language. It incorporates lightweight image processing tools that aids in editing, creating and saving images. Support for Python Imaging Library got discontinued in 2011, but a project named pillow forked the original PIL project and added Python 3.x support to it. Pillow was announced as a replacement for PIL for future usage.

The Pillow library in Python is a powerful image processing library that is used for tasks such as opening, manipulating, and saving various image file formats. It's a fork of the Python Imaging Library (PIL) and provides easy-to-use methods for common image processing tasks.

This module is not preloaded with Python. So to install it we have to execute the following command in the command-line: pip install pillow

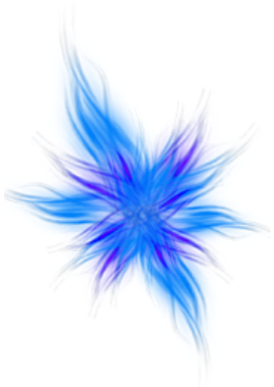
CODE:

To perform different image processing operations using the python Pillow library.

```
from PIL import Image

img = Image.open(r"test.png")
```

Output:

```
img
```



Getting Image Information:

```
print(f"{img.mode=}")
print(f"{img.size=}")
print(f"{img.format=}")
```

Output:

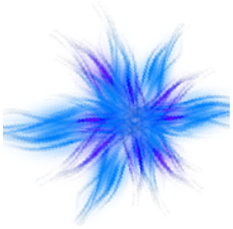
```
img.mode='RGBA'
img.size=(180, 263)
img.format='PNG'
```

Rotating an image:

```
rotated_img = img.rotate(70)
```

Output:

```
rotated_img
```



Resizing an image:

```
resized_image = img.resize((50, 50))
```

Output:

```
[19] resized_image
```

Saving the resized Image:

```
resized_image.save("resized_image.png")
```

Cropping the original image:

```
cropped_img = img.crop((130, 120, 200, 200))
```

Output:

```
[25] cropped_img
```

Writing on the image:

```
from PIL import ImageDraw

draw = ImageDraw.Draw(img)

draw.text((50, 110), "Hello World!", (255, 255, 255))
```

Output:

```
img
```

Changing resolution of the image:

```
img1 = Image.open(r"test.jpg")
img1.save("high_res.jpg", quality=95)
img1.save("medium_res.jpg", quality=25)
img1.save("low_res.jpg", quality=1)
```

Output (high_res.jpg):

```
high = Image.open(r"high_res.jpg")
high
```

Output (medium_res.jpg):

```
medium = Image.open(r"medium_res.jpg")
medium
```

Output (low_res.jpg):

```
low = Image.open(r"low_res.jpg")
low
```

Copy And Pasting Image Using Pillow

```
from PIL import Image

Image1 = Image.open('cat.jpg')

Image1copy = Image1.copy()
Image2 = Image.open('core.jpg')
Image2copy = Image2.copy()

Image1copy.paste(Image2copy, (10, 50))

Image1copy.save('pasted2.png')

imgShow = Image.open('pasted2.png')
imgShow
```

Output:

CONCLUSION : From this experiment, we learnt about the Pillow library in python which is used for image processing and to help in editing, creating and saving images. It supports a large number of image file formats including BMP, PNG, JPEG, and TIFF. In this experiment, by using pillow, we imported an image, got the information of the image, rotated the image, cropped the image, resize the image, saved the image. We also wrote text on the image and saved the image in different resolutions.