Government Engineering College

**Sec-28 Gandhinagar**

**Sem:-V (Computer Engineering Department) Subject: Python with Data Science [3150713]**

**Name :- Italiya Nirajkumar Vijaybhai**

**Er .no. : - 190130107041**

**Faclty : - BUG Mem (Binjal Gadhiya)**

**PDS Assignment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 7 | **Develop chat room applications using multithreading.(Assignment)** | Co1 |  |  |  |
| 9 | **Implement classical ciphers using python. .(Assignment)** | CO1 |  |  |  |
| 10 | **Draw graphics using Turtle. .(Assignment)** | CO1 |  |  |  |
| 11 | **Develop a program to learn GUI programming using Tkinter. .(Assignment)** | CO1 |  |  |  |

**Develop chat room applications using multithreading.(Assignment)**

Setup Server

import time, socket, sys

print('Setup Server...')

print(" Er. no. ",190130107041)

print("Italita NirajKumar \n")

time.sleep(1)

#Get the hostname, IP Address from socket and set Port

soc = socket.socket()

host\_name = socket.gethostname()

ip = socket.gethostbyname(host\_name)

port = 1234

soc.bind((host\_name, port))

print(host\_name, '({})'.format(ip))

name = input('Enter name: ')

soc.listen(1) #Try to locate using socket

print('Waiting for incoming connections...')

connection, addr = soc.accept()

print("Received connection from ", addr[0], "(", addr[1], ")\n")

print('Connection Established. Connected From: {}, ({})'.format(addr[0], addr[0]))

#get a connection from client side

client\_name = connection.recv(1024)

client\_name = client\_name.decode()

print(client\_name + ' has connected.')

print('Press [bye] to leave the chat room')

connection.send(name.encode())

while True:

message = input('Me > ')

if message == '[bye]':

message = 'Good Night...'

connection.send(message.encode())

print("\n")

break

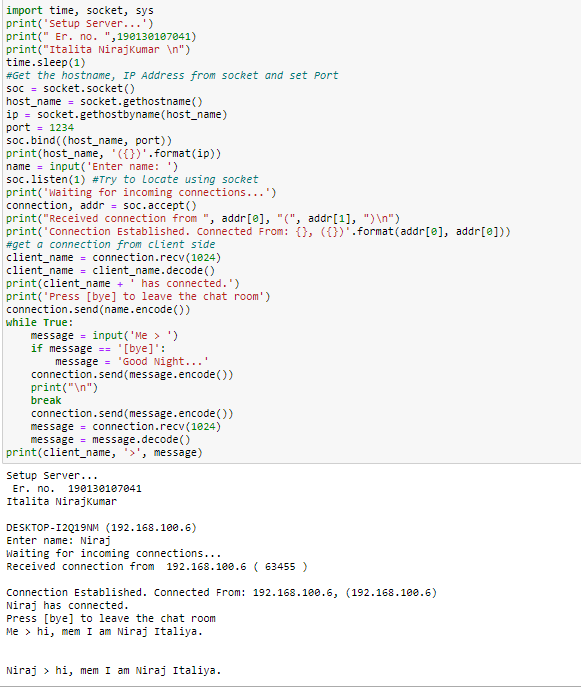
connection.send(message.encode())

message = connection.recv(1024)

message = message.decode()

print(client\_name, '>', message)

**output: -**

.

Client Server.

import time, socket, sys

print('Client Server...')

time.sleep(1)

#Get the hostname, IP Address from socket and set Port

soc = socket.socket()

shost = socket.gethostname()

ip = socket.gethostbyname(shost)

#get information to connect with the server

print(shost, '({})'.format(ip))

server\_host = input('Enter server\'s IP address:')

name = input('Enter Client\'s name: ')

port = 1234

print('Trying to connect to the server: {}, ({})'.format(server\_host, port))

time.sleep(1)

soc.connect((server\_host, port))

print("Connected...\n")

soc.send(name.encode())

server\_name = soc.recv(1024)

server\_name = server\_name.decode()

print('{} has joined...'.format(server\_name))

print('Enter [bye] to exit.')

while True:

message = soc.recv(1024)

message = message.decode()

print(server\_name, ">", message)

message = input(str("Me > "))

if message == "[bye]":

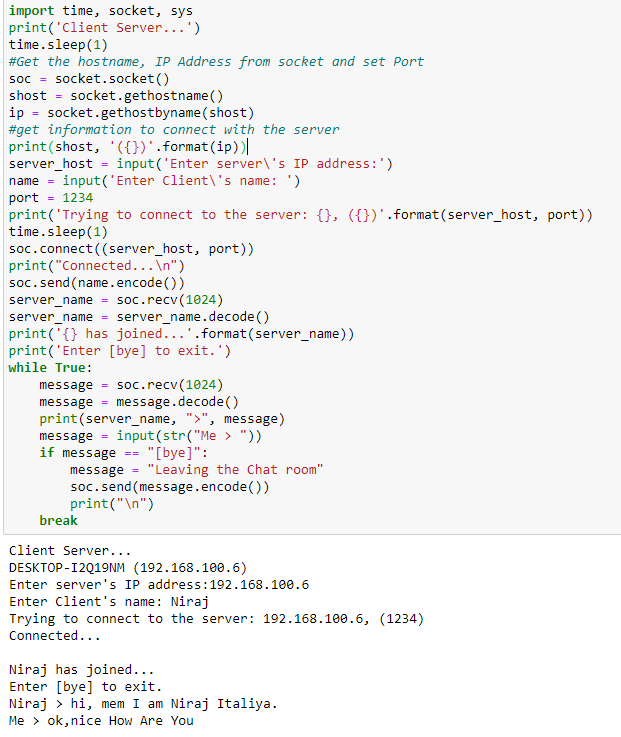
message = "Leaving the Chat room"

soc.send(message.encode())

print("\n")

break

**output: -**



**Implement classical ciphers using python**

**plaintext = input("Please enter your plaintext: ")**

**shift = input("Please enter your key: ")**

**alphabet = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"**

**ciphertext = ""**

**while isinstance(int(shift), int) == False:**

**shift = input("Please enter your key (integers only!): ")**

**shift = int(shift)**

**new\_ind = 0**

**for i in plaintext:**

**if i.lower() in alphabet:**

**new\_ind = alphabet.index(i) + shift**

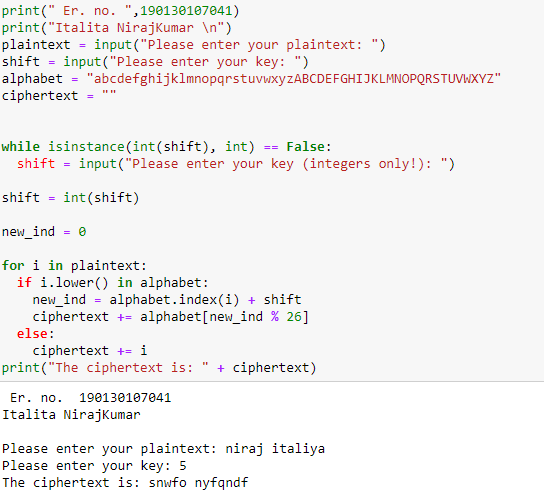
**ciphertext += alphabet[new\_ind % 26]**

**else:**

**ciphertext += i**

**print("The ciphertext is: " + ciphertext)**

**output: -**



**Draw graphics using Turtle. .(Assignment)**

**Input : -**

from turtle import \*

import random

print(190130107041)

for n in range(60):

penup()

goto(random.randint(-400, 400), random.randint(-400, 400))

pendown()

red\_amount = random.randint( 0, 30) / 100.0

blue\_amount = random.randint(50, 100) / 100.0

green\_amount = random.randint( 0, 30) / 100.0

pencolor((red\_amount, green\_amount, blue\_amount))

circle\_size = random.randint(10, 40)

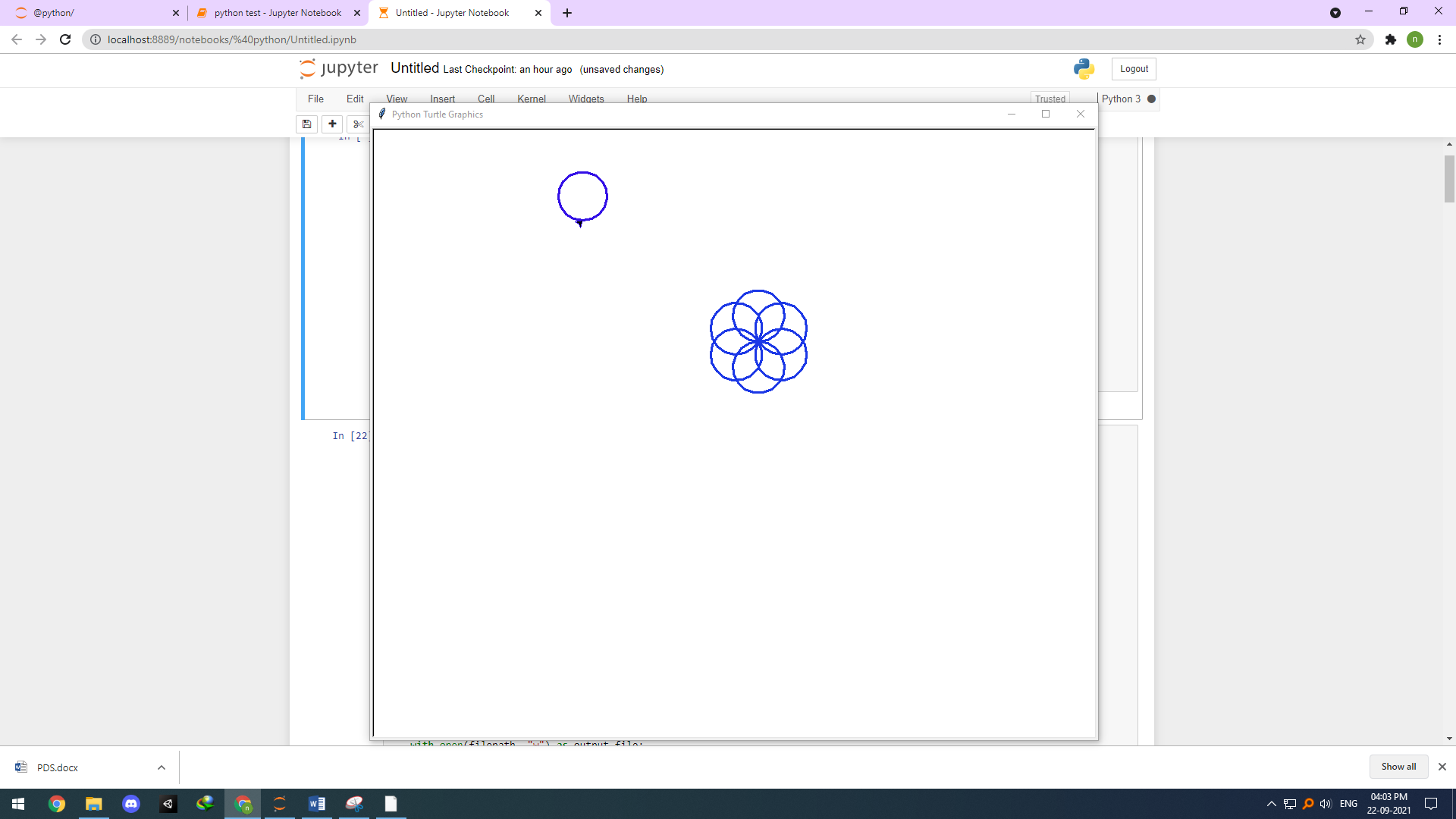
pensize(random.randint(1, 5))

for i in range(6):

circle(circle\_size)

left(60)

**output :-**



**( Video this link assignment :-** [**https://github.com/NirajItaliya/sem-5/raw/main/Untitled.mp4**](https://github.com/NirajItaliya/sem-5/raw/main/Untitled.mp4)

**)**

**Plz. Copy this link Past chorm**

**Develop a program to learn GUI programming using Tkinter.**

1. **Text Editer**

**import tkinter as tk**

**from tkinter.filedialog import askopenfilename, asksaveasfilename**

**def open\_file():**

**filepath = askopenfilename(**

**filetypes=[("Text Files", "\*.txt"), ("All Files", "\*.\*")]**

**)**

**if not filepath:**

**return**

**txt\_edit.delete(1.0, tk.END)**

**with open(filepath, "r") as input\_file:**

**text = input\_file.read()**

**txt\_edit.insert(tk.END, text)**

**window.title(f"Simple Text Editor - {filepath}")**

**def save\_file():**

**filepath = asksaveasfilename(**

**defaultextension="txt",**

**filetypes=[("Text Files", "\*.txt"), ("All Files", "\*.\*")],**

**)**

**if not filepath:**

**return**

**with open(filepath, "w") as output\_file:**

**text = txt\_edit.get(1.0, tk.END)**

**output\_file.write(text)**

**window.title(f"Simple Text Editor - {filepath}")**

**window = tk.Tk()**

**window.title("Simple Text Editor")**

**window.rowconfigure(0, minsize=800, weight=1)**

**window.columnconfigure(1, minsize=800, weight=1)**

**txt\_edit = tk.Text(window)**

**fr\_buttons = tk.Frame(window, relief=tk.RAISED, bd=2)**

**btn\_open = tk.Button(fr\_buttons, text="Open", command=open\_file)**

**btn\_save = tk.Button(fr\_buttons, text="Save As...", command=save\_file)**

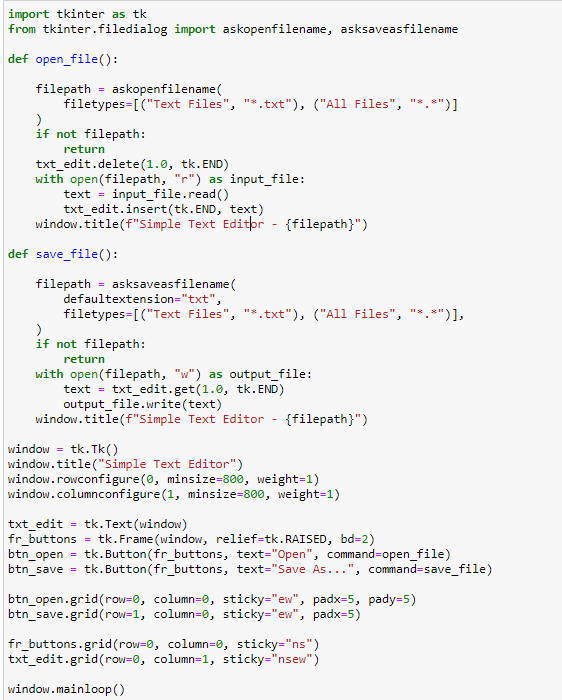
**btn\_open.grid(row=0, column=0, sticky="ew", padx=5, pady=5)**

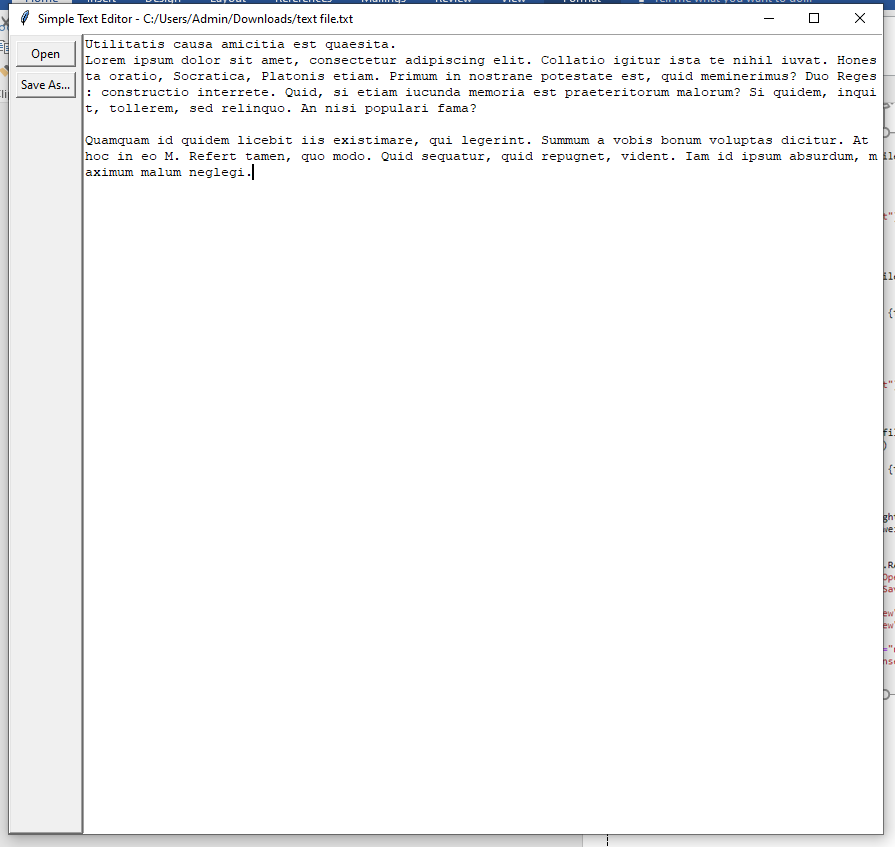
**btn\_save.grid(row=1, column=0, sticky="ew", padx=5)**

**fr\_buttons.grid(row=0, column=0, sticky="ns")**

**txt\_edit.grid(row=0, column=1, sticky="nsew")**

**window.mainloop()**



**out** 

( This practicle Viedo Link :- <https://github.com/NirajItaliya/sem-5/raw/main/Note%20Pad.mp4> )

1. **Paint**

# paint

from tkinter import \*

from tkinter.colorchooser import askcolor

class Paint(object):

DEFAULT\_PEN\_SIZE = 5.0

DEFAULT\_COLOR = 'black'

def \_\_init\_\_(self):

self.root = Tk()

self.pen\_button = Button(self.root, text='pen', command=self.use\_pen)

self.pen\_button.grid(row=0, column=0)

self.brush\_button = Button(self.root, text='brush', command=self.use\_brush)

self.brush\_button.grid(row=0, column=1)

self.color\_button = Button(self.root, text='color', command=self.choose\_color)

self.color\_button.grid(row=0, column=2)

self.eraser\_button = Button(self.root, text='eraser', command=self.use\_eraser)

self.eraser\_button.grid(row=0, column=3)

self.choose\_size\_button = Scale(self.root, from\_=1, to=10, orient=HORIZONTAL)

self.choose\_size\_button.grid(row=0, column=4)

self.c = Canvas(self.root, bg='white', width=600, height=600)

self.c.grid(row=1, columnspan=5)

self.setup()

self.root.mainloop()

def setup(self):

self.old\_x = None

self.old\_y = None

self.line\_width = self.choose\_size\_button.get()

self.color = self.DEFAULT\_COLOR

self.eraser\_on = False

self.active\_button = self.pen\_button

self.c.bind('<B1-Motion>', self.paint)

self.c.bind('<ButtonRelease-1>', self.reset)

def use\_pen(self):

self.activate\_button(self.pen\_button)

def use\_brush(self):

self.activate\_button(self.brush\_button)

def choose\_color(self):

self.eraser\_on = False

self.color = askcolor(color=self.color)[1]

def use\_eraser(self):

self.activate\_button(self.eraser\_button, eraser\_mode=True)

def activate\_button(self, some\_button, eraser\_mode=False):

self.active\_button.config(relief=RAISED)

some\_button.config(relief=SUNKEN)

self.active\_button = some\_button

self.eraser\_on = eraser\_mode

def paint(self, event):

self.line\_width = self.choose\_size\_button.get()

paint\_color = 'white' if self.eraser\_on else self.color

if self.old\_x and self.old\_y:

self.c.create\_line(self.old\_x, self.old\_y, event.x, event.y,

width=self.line\_width, fill=paint\_color,

capstyle=ROUND, smooth=TRUE, splinesteps=36)

self.old\_x = event.x

self.old\_y = event.y

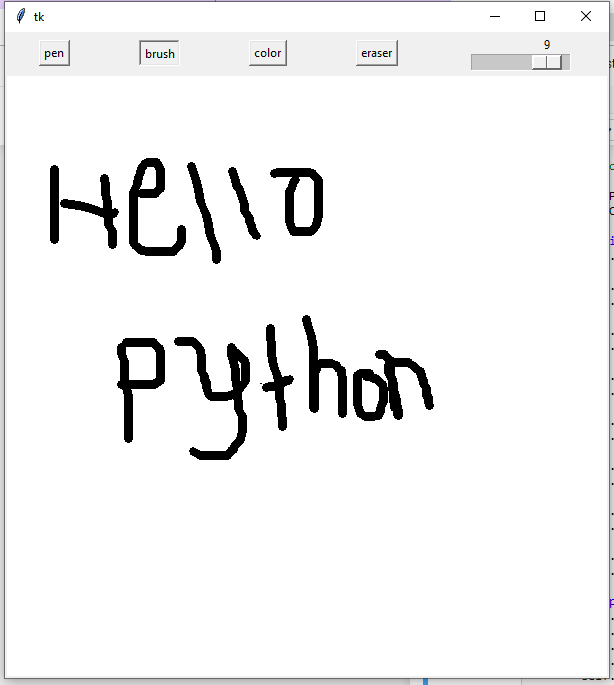
def reset(self, event):

self.old\_x, self.old\_y = None, None

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

Paint()

**Output:-**



1. **Photoediter**

from tkinter import \*

from tkinter import ttk

from tkinter import filedialog

from tkinter.filedialog import askopenfilename,asksaveasfilename

from PIL import Image, ImageTk, ImageFilter, ImageEnhance, ImageOps

import os

# contrast border thumbnail

root = Tk()

root.title("Simple Photo Editor")

root.geometry("640x640")

# create functions

def selected():

global img\_path, img

img\_path = filedialog.askopenfilename(initialdir=os.getcwd())

img = Image.open(img\_path)

img.thumbnail((350, 350))

#imgg = img.filter(ImageFilter.BoxBlur(0))

img1 = ImageTk.PhotoImage(img)

canvas2.create\_image(300, 210, image=img1)

canvas2.image=img1

def blur(event):

global img\_path, img1, imgg

for m in range(0, v1.get()+1):

img = Image.open(img\_path)

img.thumbnail((350, 350))

imgg = img.filter(ImageFilter.BoxBlur(m))

img1 = ImageTk.PhotoImage(imgg)

canvas2.create\_image(300, 210, image=img1)

canvas2.image=img1

def brightness(event):

global img\_path, img2, img3

for m in range(0, v2.get()+1):

img = Image.open(img\_path)

img.thumbnail((350, 350))

imgg = ImageEnhance.Brightness(img)

img2 = imgg.enhance(m)

img3 = ImageTk.PhotoImage(img2)

canvas2.create\_image(300, 210, image=img3)

canvas2.image=img3

def contrast(event):

global img\_path, img4, img5

for m in range(0, v3.get()+1):

img = Image.open(img\_path)

img.thumbnail((350, 350))

imgg = ImageEnhance.Contrast(img)

img4 = imgg.enhance(m)

img5 = ImageTk.PhotoImage(img4)

canvas2.create\_image(300, 210, image=img5)

canvas2.image=img5

def rotate\_image(event):

global img\_path, img6, img7

img = Image.open(img\_path)

img.thumbnail((350, 350))

img6 = img.rotate(int(rotate\_combo.get()))

img7 = ImageTk.PhotoImage(img6)

canvas2.create\_image(300, 210, image=img7)

canvas2.image=img7

def flip\_image(event):

global img\_path, img8, img9

img = Image.open(img\_path)

img.thumbnail((350, 350))

if flip\_combo.get() == "FLIP LEFT TO RIGHT":

img8 = img.transpose(Image.FLIP\_LEFT\_RIGHT)

elif flip\_combo.get() == "FLIP TOP TO BOTTOM":

img8 = img.transpose(Image.FLIP\_TOP\_BOTTOM)

img9 = ImageTk.PhotoImage(img8)

canvas2.create\_image(300, 210, image=img9)

canvas2.image=img9

def image\_border(event):

global img\_path, img10, img11

img = Image.open(img\_path)

img.thumbnail((350, 350))

img10 = ImageOps.expand(img, border=int(border\_combo.get()), fill=95)

img11 = ImageTk.PhotoImage(img10)

canvas2.create\_image(300, 210, image=img11)

canvas2.image=img11

img1 = None

img3 = None

img5 = None

img7 = None

img9 = None

img11 = None

def save():

global img\_path, imgg, img1, img2, img3, img4, img5, img6, img7, img8, img9, img10, img11

#file=None

ext = img\_path.split(".")[-1]

file=asksaveasfilename(defaultextension =f".{ext}",filetypes=[("All Files","\*.\*"),("PNG file","\*.png"),("jpg file","\*.jpg")])

if file:

if canvas2.image==img1:

imgg.save(file)

elif canvas2.image==img3:

img2.save(file)

elif canvas2.image==img5:

img4.save(file)

elif canvas2.image==img7:

img6.save(file)

elif canvas2.image==img9:

img8.save(file)

elif canvas2.image==img11:

img10.save(file)

# create labels, scales and comboboxes

blurr = Label(root, text="Blur:", font=("ariel 17 bold"), width=9, anchor='e')

blurr.place(x=15, y=8)

v1 = IntVar()

scale1 = ttk.Scale(root, from\_=0, to=10, variable=v1, orient=HORIZONTAL, command=blur)

scale1.place(x=150, y=10)

bright = Label(root, text="Brightness:", font=("ariel 17 bold"))

bright.place(x=8, y=50)

v2 = IntVar()

scale2 = ttk.Scale(root, from\_=0, to=10, variable=v2, orient=HORIZONTAL, command=brightness)

scale2.place(x=150, y=55)

contrast = Label(root, text="Contrast:", font=("ariel 17 bold"))

contrast.place(x=35, y=92)

v3 = IntVar()

scale3 = ttk.Scale(root, from\_=0, to=10, variable=v3, orient=HORIZONTAL, command=contrast)

scale3.place(x=150, y=100)

rotate = Label(root, text="Rotate:", font=("ariel 17 bold"))

rotate.place(x=370, y=8)

values = [0, 90, 180, 270, 360]

rotate\_combo = ttk.Combobox(root, values=values, font=('ariel 10 bold'))

rotate\_combo.place(x=460, y=15)

rotate\_combo.bind("<<ComboboxSelected>>", rotate\_image)

flip = Label(root, text="Flip:", font=("ariel 17 bold"))

flip.place(x=400, y=50)

values1 = ["FLIP LEFT TO RIGHT", "FLIP TOP TO BOTTOM"]

flip\_combo = ttk.Combobox(root, values=values1, font=('ariel 10 bold'))

flip\_combo.place(x=460, y=57)

flip\_combo.bind("<<ComboboxSelected>>", flip\_image)

border = Label(root, text="Add border:", font=("ariel 17 bold"))

border.place(x=320, y=92)

values2 = [i for i in range(10, 45, 5)]

border\_combo = ttk.Combobox(root, values=values2, font=("ariel 10 bold"))

border\_combo.place(x=460, y=99)

border\_combo.bind("<<ComboboxSelected>>", image\_border)

# create canvas to display image

canvas2 = Canvas(root, width="600", height="420", relief=RIDGE, bd=2)

canvas2.place(x=15, y=150)

# create buttons

btn1 = Button(root, text="Select Image", bg='black', fg='gold', font=('ariel 15 bold'), relief=GROOVE, command=selected)

btn1.place(x=100, y=595)

btn2 = Button(root, text="Save", width=12, bg='black', fg='gold', font=('ariel 15 bold'), relief=GROOVE, command=save)

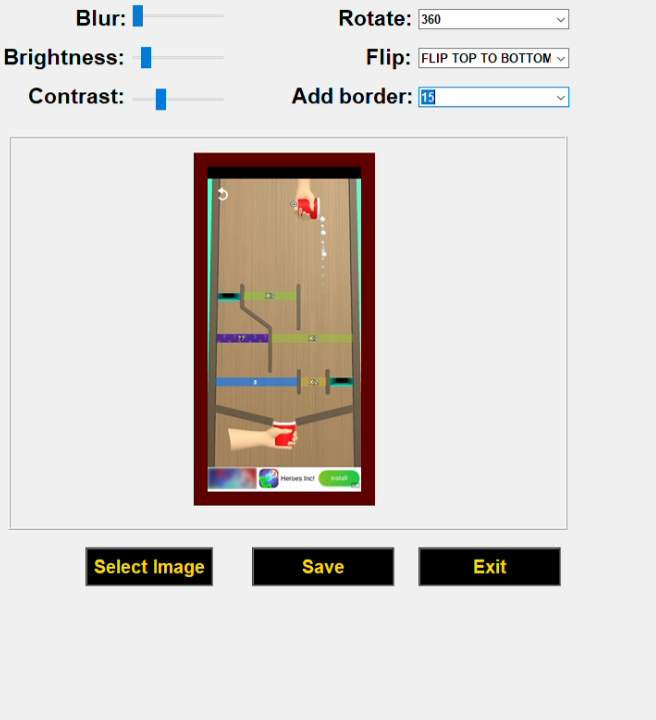
btn2.place(x=280, y=595)

btn3 = Button(root, text="Exit", width=12, bg='black', fg='gold', font=('ariel 15 bold'), relief=GROOVE, command=root.destroy)

btn3.place(x=460, y=595)

root.mainloop()

**output :-**



( This practicle Viedo Link :- <https://github.com/NirajItaliya/sem-5/raw/main/Paint.mp4> )