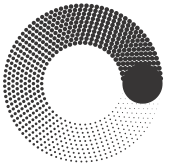
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № \_1\_**

**Дисциплина:** \_Распознавание образов в информационных и автоматизированных системах\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Тема:** \_\_\_ Работа с цветами\_\_\_\_\_

**Выполнил: студент группы \_211-723\_\_\_**

\_\_\_\_\_Барицкий Владимир Денисович\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_26.11.22\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2022**

**Цель:**

Целью данной работы является изучение базовых операций над цветовыми каналами

изображений и реализация некоторых фильтров на их основе.

**Задание:**

Необходимо разработать приложение Windows Forms, способное осуществлять:

1. загрузку и отображение двух изображений по выбору пользователя;

2. возможность применения базовых операций к загруженным изображениям;

3. возможность применения оконных и комбинированных фильтров к загруженным

изображениям

**Листинг кода:**

import cv2

import numpy as np

from PIL import Image, ImageEnhance, ImageDraw, ImageTk

from tkinter import \*

from tkinter.ttk import Progressbar, Entry

from PIL import \*

class Image\_(Frame):

img1 = cv2.imread('ball.jpg', cv2.IMREAD\_COLOR)

def Copy(self):

original = cv2.imread('ball.jpg', cv2.IMREAD\_COLOR)

self.img1=original

def set\_hsv(self):

def show\_message():

self.value1 = int(entry.get())

self.value2 = int(entry1.get())

self.value3 = int(entry2.get())

self.value4 = int(entry3.get())

self.value5 = int(entry4.get())

self.value6 = int(entry5.get())

self.HSV()

root = Tk()

root.title("Настройки")

root.geometry("250x500")

lbl = Label(root, text = "Channel HMin")

lbl.pack(anchor=CENTER, padx=6, pady=6)

entry = Entry(root)

entry.pack(anchor=CENTER, padx=6, pady=6)

lbl3 = Label(root, text = "Channel HMax")

lbl3.pack(anchor=CENTER, padx=6, pady=6)

entry3= Entry(root)

entry3.pack(anchor=CENTER, padx=6, pady=6)

lbl1 = Label(root, text = "Channel SMin")

lbl1.pack(anchor=CENTER, padx=6, pady=6)

entry1 = Entry(root)

entry1.pack(anchor=CENTER, padx=6, pady=6)

lbl4 = Label(root, text = "Channel SMax")

lbl4.pack(anchor=CENTER, padx=6, pady=6)

entry4 = Entry(root)

entry4.pack(anchor=CENTER, padx=6, pady=6)

lbl2 = Label(root, text = "Channel VMin")

lbl2.pack(anchor=CENTER, padx=6, pady=6)

entry2 = Entry(root)

entry2.pack(anchor=CENTER, padx=6, pady=6)

lbl5 = Label(root, text = "Channel VMax")

lbl5.pack(anchor=CENTER, padx=6, pady=6)

entry5 = Entry(root)

entry5.pack(anchor=CENTER, padx=6, pady=6)

btn = Button(root, text="Click", command=show\_message)

btn.pack(anchor=CENTER, padx=6, pady=6)

root.mainloop()

def Settings(self):

root = Tk()

l1 = Label(root)

l2 = Label(root)

def mv(event):

l1["text"] = "Минимальное пороговое значение: " + str(int(scale.get()))

l2["text"] = "Максимальное пороговое значение: " + str(int(scale1.get()))

self.v1 = int(scale.get())

self.v2 = int(scale1.get())

scale = Scale( root, from\_=0, to=255, orient=HORIZONTAL, command = mv)

scale.pack(anchor = CENTER)

scale1 = Scale( root, from\_=0, to=255, orient=HORIZONTAL, command = mv)

scale1.pack(anchor=CENTER)

l1.pack()

l2.pack()

button = Button(root, text="Обработать изображение", command=self.Cartoon)

button.pack(anchor=CENTER)

root.mainloop()

def Black\_White(img):

thresh = 155

img = cv2.imread('ball.jpg', cv2.IMREAD\_GRAYSCALE)

# threshold the image

img\_binary = cv2.threshold(img, thresh, 255, cv2.THRESH\_BINARY)[1]

cv2.imshow("BlackWhite", img\_binary)

cv2.waitKey(0)

def Cartoon(self):

edges1 = cv2.bitwise\_not(cv2.Canny(self.img1, self.v1, self.v2))

dst = cv2.edgePreservingFilter(self.img1, flags=2, sigma\_s=64, sigma\_r=0.25)

cartoon1 = cv2.bitwise\_and(dst, dst, mask=edges1)

imghstack = np.hstack((self.img1, cartoon1))

cv2.imshow("Cartoon1", imghstack)

cv2.waitKey(0)

cv2.destroyAllWindows()

def Channel\_S(self):

lower = 80

upper = 150

Channel(lower, upper)

def Channel\_K(self):

lower = 0

upper = 40

Channel(lower, upper)

def Channel\_Z(self):

lower = 40

upper = 80

Channel(lower, upper)

def Blur(self):

image = cv2.imread("love.jpg")

gaus = cv2.medianBlur(image, 15)

gaus1 = cv2.resize(gaus,(500,500))

cv2.imshow("Gaussian blur", gaus1)

cv2.waitKey(0)

def Mat(self):

root = Tk()

root.title("Настройки")

root.geometry("265x250")

def show\_message():

self.k1 = int(entry.get())

self.k2 = int(entry1.get())

self.k3 = int(entry2.get())

self.k4 = int(entry3.get())

self.k5 = int(entry4.get())

self.k6 = int(entry5.get())

self.k7 = int(entry6.get())

self.k8 = int(entry7.get())

self.k9 = int(entry8.get())

self.Mat\_filter()

entry = Entry(root)

entry.place(x=60, y = 30, width = 30, height = 30)

entry1 = Entry(root)

entry1.place(x=120, y = 30, width = 30, height = 30)

entry2 = Entry(root)

entry2.place(x=180, y = 30, width = 30, height = 30)

entry3 = Entry(root)

entry3.place(x=60, y = 90, width = 30, height = 30)

entry4 = Entry(root)

entry4.place(x=120, y = 90, width = 30, height = 30)

entry5 = Entry(root)

entry5.place(x=180, y = 90, width = 30, height = 30)

entry6 = Entry(root)

entry6.place(x=60, y = 150, width = 30, height = 30)

entry7 = Entry(root)

entry7.place(x=120, y = 150, width = 30, height = 30)

entry8 = Entry(root)

entry8.place(x=180, y = 150, width = 30, height = 30)

btn = Button(root, text="Click", command=show\_message)

btn.place(x=80, y = 200, width = 100, height = 30)

root.mainloop()

def Mat\_filter(self):

kernel = np.array([[self.k1,self.k2,self.k3], [self.k4,self.k5,self.k6], [self.k7,self.k8,self.k9]])

im = cv2.filter2D(self.img1, -1, kernel)

imghstack = np.hstack((self.img1, im))

cv2.imshow('MyPhoto', imghstack )

cv2.waitKey(0)

cv2.destroyAllWindows()

def Crossing(self):

image = cv2.imread("love.jpg")

image = cv2.resize(image,(500,500))

M = np.ones(image.shape,dtype="uint8")\*50

added = cv2.bitwise\_and (image, M)

imghstack = np.hstack((image, added))

cv2.imshow("Пересечение", imghstack)

def Not(self):

img = cv2.imread("original\_fapiao.png")

img2 = cv2.imread("extract\_fapiao.png")

bitwiseNot = cv2.bitwise\_not(img2)

bitwiseNot = cv2.resize(bitwiseNot,(400, 700))

img = cv2.resize(img,(400, 700))

imghstack = np.hstack((img, bitwiseNot))

cv2.imshow("Not",imghstack)

cv2.waitKey(0)

cv2.destroyAllWindows()

def Xor(self):

img = cv2.imread("p1.png")

img = cv2.resize(img,(500, 200))

img2 = cv2.imread("p2.png")

img2 = cv2.resize(img2,(500, 200))

bitwiseXor = cv2.bitwise\_xor(img,img2)

imghstack = np.hstack((img, img2, bitwiseXor))

cv2.imshow ("bitwiseXor XOR операция:", imghstack)

cv2.waitKey(0)

cv2.destroyAllWindows()

def Contrast(self, factor):

image = Image.open("ball.jpg")

enhancer = ImageEnhance.Contrast(image)

im\_output = enhancer.enhance(factor)

im\_output.save("ans.jpg", "JPEG")

image = cv2.imread("ans.jpg")

image2 = cv2.imread("smechariki.jpg")

imghstack = np.hstack((image2, image))

cv2.imshow("Contrast", imghstack)

def Aqua(self):

image = Image.open("love.jpg")

enhancer = ImageEnhance.Contrast(image)

im\_output = enhancer.enhance(8)

im\_output.save("girl1.jpg", "JPEG")

image = cv2.imread("girl1.jpg")

img2 = cv2.imread("aqua1.jpg")

gaus = cv2.medianBlur(image, 17)

gaus1 = cv2.resize(gaus,(500,500))

aqua = cv2.resize(img2,(500,500))

combine = cv2.addWeighted(gaus1,0.5,aqua,0.4,0)

cv2.imshow('combine',combine)

cv2.waitKey(0)

def HSV(self):

img\_hsv= self.img1

hsv = cv2.cvtColor(img\_hsv, cv2.COLOR\_BGR2HSV)

h, s, v = cv2.split(hsv)

h1, s1, v1, h2, s2, v2 = self.value1, self.value2, self.value3, self.value4, self.value5, self.value6

mask = cv2.inRange(hsv,(h1, s1, v1), (h2, s2, v2) )

cv2.imshow("HSV",mask)

def Bright(self, value):

image = Image.open("ball.jpg")

draw = ImageDraw.Draw(image)

width = image.size[0] #Определяем ширину.

height = image.size[1] #Определяем высоту.

pix = image.load()

for i in range(width):

for j in range(height):

a = pix[i, j][0] + value

b = pix[i, j][1] + value

c = pix[i, j][2] + value

if (a < 0):

a = 0

if (b < 0):

b = 0

if (c < 0):

c = 0

if (a > 255):

a = 255

if (b > 255):

b = 255

if (c > 255):

c = 255

draw.point((i, j), (a, b, c))

image.save("ans.jpg", "JPEG")

del draw

image = cv2.imread("ans.jpg")

image1 = cv2.imread("smechariki.jpg")

imgstack = np.hstack((image1, image))

cv2.imshow("Bright", imgstack)

def Sepia(self):

original = cv2.imread("ball.jpg")

img1 = cv2.imread("ball.jpg")

img1 = np.array(img1, dtype=np.float64)

img1 = cv2.transform(img1, np.matrix([[0.272, 0.534, 0.131],

[0.349, 0.686, 0.168],

[0.393, 0.769, 0.189]]))

img1[np.where(img1 > 255)] = 255

img1 = np.array(img1, dtype=np.uint8)

cv2.imshow("original", original)

cv2.imshow("Output", img1)

cv2.waitKey(0)

cv2.destroyAllWindows()

def Channel(l, u):

def hsv(img, l, u):

hsv = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV)

lower = np.array([l,128,128])

upper = np.array([u,255,255])

mask = cv2.inRange(hsv, lower, upper)

return mask

img = cv2.imread('ball1.jfif')

original = img.copy()

res = np.zeros(img.shape, np.uint8)

mask = hsv(img, l, u)

inv\_mask = cv2.bitwise\_not(mask)

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

res1 = cv2.bitwise\_and(img, img, mask= mask)

res2 = cv2.bitwise\_and(gray, gray, mask= inv\_mask)

for i in range(3):

res[:, :, i] = res2

img = cv2.bitwise\_or(res1, res)

imghstack = np.hstack((original, img))

cv2.imshow("Channel",imghstack)

cv2.waitKey(0)

cv2.destroyAllWindows()

def menu\_bc():

window1 = Tk()

window1.title("Меню яркости и контраста")

window1.geometry("300x150")

window1.configure(bg='#D0FBFF')

lbl = Label(window1, text = "Выберите, с чем хотите работать")

lbl.pack()

def dauther1():

val = 0

img = Image\_()

def change1():

dauther1.val = 60

def change2():

dauther1.val = 0

def change3():

dauther1.val = -60

win1 = Toplevel(window1)

Plus = Button(win1, text="Увеличить яркость", command=change1)

Normal = Button(win1, text="Обычная яркость", command=change2)

Minus = Button(win1, text="Уменьшить яркость", command=change3)

button = Button(win1, text="Изменить",bg="#eec6ea",command=lambda: img.Bright(dauther1.val))

Plus.pack(anchor=CENTER, padx = 20, pady = 5)

Normal.pack(anchor=CENTER, padx = 20, pady = 5)

Minus.pack(anchor=CENTER, padx = 20, pady = 5)

button.pack(anchor=CENTER, padx = 20, pady = 20)

def dauther2():

val = 0

img = Image\_()

def change1():

dauther2.val = 1.5

def change2():

dauther2.val = 1

def change3():

dauther2.val = 0.5

win1 = Toplevel(window1)

Plus = Button(win1, text="Увеличить контрастность", command=change1)

Normal = Button(win1, text="Обычная контрастность", command=change2)

Minus = Button(win1, text="Уменьшить контрастность", command=change3)

button = Button(win1, text="Изменить",bg="#eec6ea",command=lambda: img.Contrast(dauther2.val))

Plus.pack(anchor=CENTER, padx = 20, pady = 5)

Normal.pack(anchor=CENTER, padx = 20, pady = 5)

Minus.pack(anchor=CENTER, padx = 20, pady = 5)

button.pack(anchor=CENTER, padx = 20, pady = 20)

btn1 = Button(window1, text= "Яркость", command = dauther1)

btn1.place(x = 50, y = 60, width=80, height=30)

btn2 = Button(window1, text= "Контраст", command = dauther2)

btn2.place(x = 180, y = 60, width=80, height=30)

window1.mainloop()

def clicked3():

im = Image\_()

i1 = im.Black\_White()

imghstack = np.hstack(im.img1, i1)

cv2.namedWindow("BlackWhite")

cv2.imshow("BlackWhite", imghstack)

cv2.waitKey(0)

def clicked2():

img = Image\_()

img.Blur()

def clicked4():

exit()

def menu\_log():

root = Tk()

im = Image\_()

root.title("choice operation")

w = root.winfo\_screenwidth()

h = root.winfo\_screenheight()

w = w//2 # середина экрана

h = h//2

w = w - 200 # смещение от середины

h = h - 200

root.geometry('400x250+{}+{}'.format(w, h))

root.configure(bg='#D0FBFF')

btn = Button(root, text="Пересечение", padx=5, pady=5, command = im.Crossing, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

btn = Button(root, text="Дополнение", padx=5, pady=5, command = im.Not, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

btn = Button(root, text="Исключение", padx=5, pady=5, command = im.Xor, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

root.mainloop()

def menu\_ch():

root = Tk()

im = Image\_()

root.title("choice channel")

w = root.winfo\_screenwidth()

h = root.winfo\_screenheight()

w = w//2 # середина экрана

h = h//2

w = w - 200 # смещение от середины

h = h - 200

root.geometry('400x250+{}+{}'.format(w, h))

root.configure(bg='#D0FBFF')

btn = Button(root, text="Вывод синего канала", padx=5, pady=5, command = im.Channel\_S, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

btn = Button(root, text="Вывод красного канала", padx=5, pady=5, command = im.Channel\_K, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

btn = Button(root, text="Вывод зеленого канала", padx=5, pady=5, command = im.Channel\_Z, bg='#7CFFA8')

btn.pack(anchor="center", padx=20, pady=20)

root.mainloop()

def Menu():

window = Tk()

img = Image\_()

window.title("Menu")

w = window.winfo\_screenwidth()

h = window.winfo\_screenheight()

w = w//2 # середина экрана

h = h//2

w = w - 200 # смещение от середины

h = h - 200

window.geometry('600x300+{}+{}'.format(w, h))

window.config(background="#D0FBFF")

btn = Button(window, text="Черно-белая версия фото", padx=5, pady=5, command =img.Black\_White , bg="#7CFFA8" )

btn.place(x=70, y = 40, width=200, height=30)

btn1 = Button(window, text="Сепия версия", padx=5, pady=5, command = img.Sepia, bg="#7CFFA8" )

btn1.place(x=310, y = 40, width=200, height=30)

btn10 = Button(window, text="Логические операции", padx=5, pady=5, command =menu\_log , bg="#7CFFA8" )

btn10.place(x=70, y = 100, width=200, height=30)

btn3 = Button(window, text="Размытие", padx=5, pady=5, command = clicked2, bg="#7CFFA8" )

btn3.place(x=290, y = 100, width=100, height=30)

btn7 = Button(window, text="Cartoon filter", padx=5, pady=5, command = img.Settings, bg='#7CFFA8')

btn7.place(x=410, y = 100, width=100, height=30)

btn5 = Button(window, text="Вывод канала", padx=5, pady=5, command = menu\_ch,bg="#7CFFA8")

btn5.place(x=70, y = 160, width=100, height=30)

btn6 = Button(window, text="Изменение яркости и контраста", padx=5, pady=5, command =menu\_bc ,bg="#7CFFA8" )

btn6.place(x=190, y = 160, width=200, height=30)

btn2 = Button(window, text="HSV", padx=5, pady=5, command = img.set\_hsv, bg="#7CFFA8" )

btn2.place(x=410, y = 160, width=100, height=30)

btn8 = Button(window, text="Оконный фильтр", padx=5, pady=5, command =img.Mat , bg='#7CFFA8')

btn8.place(x=70, y = 220, width=150, height=30)

btn9 = Button(window, text="Акварельный фильтр", padx=5, pady=5, command =img.Aqua , bg='#7CFFA8')

btn9.place(x=255, y = 220, width=150, height=30)

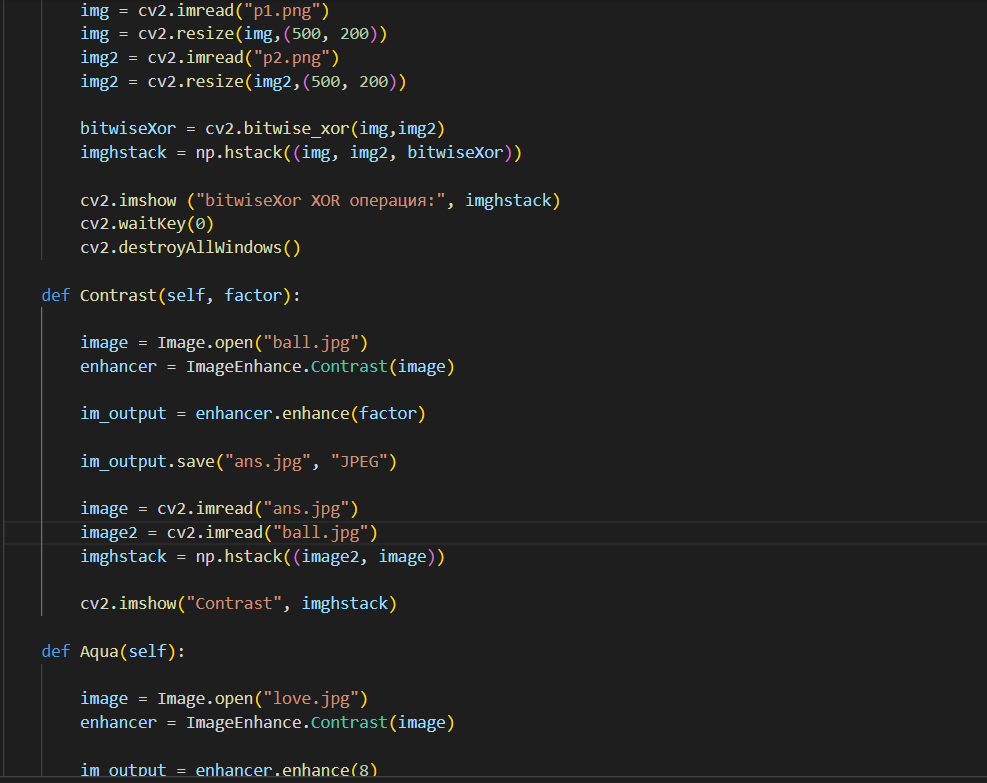
btn4 = Button(window, text="Выход", padx=5, pady=5, command = clicked4, bg='#7CFFA8')

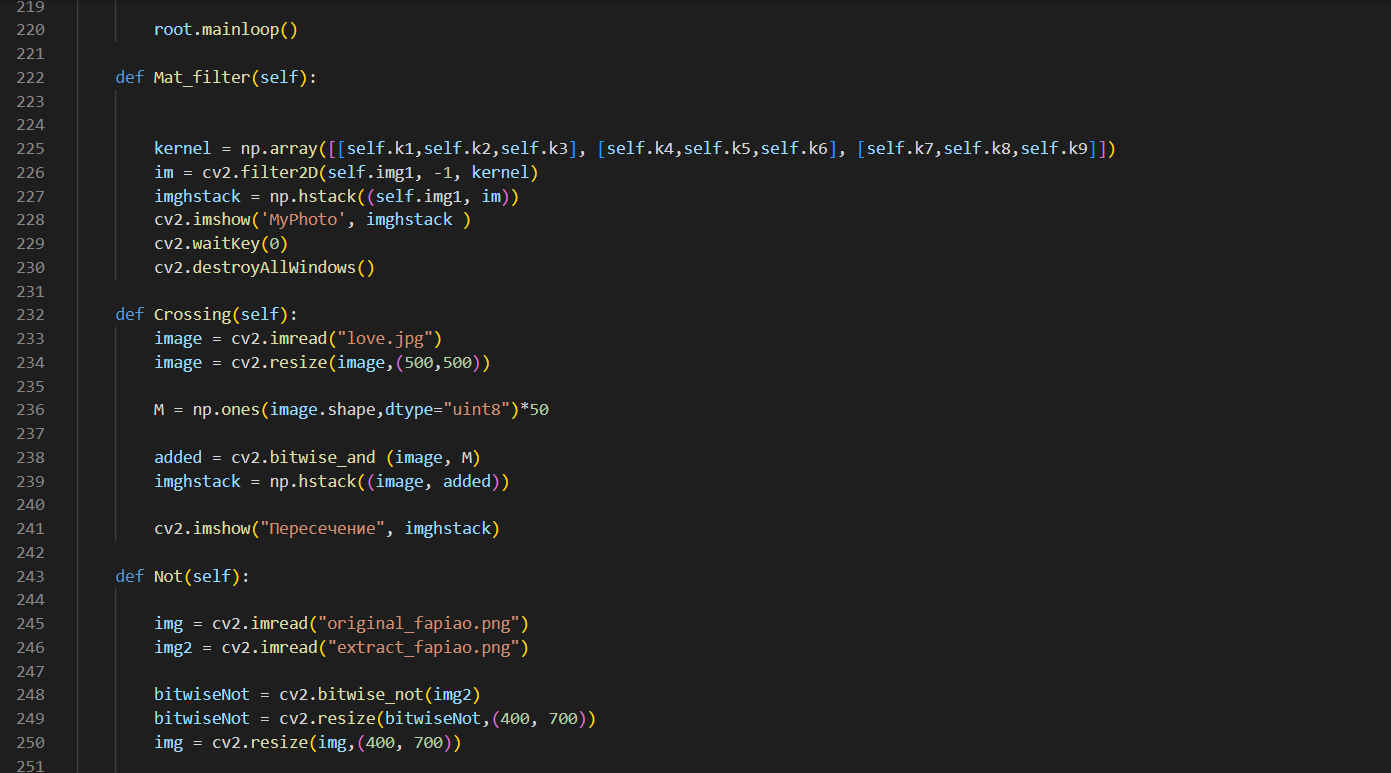
btn4.place(x=440, y = 220, width=70, height=30)

window.mainloop()

Menu()

**Скриншоты кода:**





**Скриншоты работы кода:**



