МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Информатика»

Тема: Введение в архитектуру компьютера

Студент гр. 3344 _____ Валиев Р.А.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2023

Цель работы

Освоение обработки изображений на языке Python.

Задание.

Вариант 1

Предстоит решить 3 подзадачи, используя библиотеку Pillow (PIL). Для реализации требуемых функций студент должен использовать numpy и PIL. Аргумент image в функциях подразумевает объект типа <class 'PIL.Image.Image'>

1) Рисование треугольника

Необходимо написать функцию triangle(), которая рисует на изображении треугольник

Функция triangle() принимает на вход:

- *Изображение (img)*
- *Координаты вершин (x0,y0,x1,y1,x2,y2)*
- *Толщину линий (thickness)*
- *Цвет линий (color) - представляет собой список (list) из 3-х целых чисел*
- *Цвет, которым залит (fill_color - если значение None, значит треугольник не залит) - представляет собой список (list) из 3-х целых чисел*

Функция должна вернуть исходное обработанное изображение.

2) Замена наиболее часто встречаемого цвета.

Необходимо написать функцию change_color(), которая заменяет наиболее часто встречаемый цвет на переданный.

Функция change_color() принимает на вход:

- *Изображение (img)*
- *Цвет (color - представляет собой список из трех целых чисел)*

Функция должна найти в изображении самый частый цвет и заменить его на переданный, затем вернуть новое изображение (исходное изображение не должно меняться).

3) Коллаж

Необходимо написать функцию collage().

Функция collage() принимает на вход:

- *Изображение (img)*
- *Количество изображений по "оси" Y (N - натуральное)*
- *Количество изображений по "оси" X (M - натуральное)*

Выполнение работы

Была реализована функция triangle(),которая принимает изображение,координаты вершин треугольника, толщину линий, цвет линий и цвет заливки. Она использует библиотеку PIL для рисования треугольника на изображении. Если задан цвет заливки то треугольник будет залит этим цветом, иначе только линии треугольника.

Функция change_color(), принимает изображение и цвет для замены. Она находит наиболее часто встречаемый цвет в изображении и заменяет его на заданный цвет. Для этого функция использует библиотеку Pillow.

Функция collage() создает коллаж изображений, где исходное изображение повторяется заданное количество раз по высоте и ширине.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	Img=Image.new(“RGB”, (80,80),”red”)triangle(img,30,45,37,28,50,43,20, [7,47,23], [123,33,127]).show()	img	-
2.	img=Image.new(“RGB”, (80,80),”red”) change_color(triangle(img,30,45,37,28,50,43,20, [7,47,23],color = [25,43,120]).show()	img	-
3.	mix(Image.open("krab1.jpeg"), {0:2,1:2,2:2,3:5,4:5,5:5,6:8,7:8,8:8})img=Image.new(“RGB”,(80,80),”red”) collage(triangle(img,30,45,37,28,50,43,20,[7,47,23],color = [123,33,127]).show()	img	-

Выводы

Была освоена обработка изображений на языке Python. Были получены базовые навыки работы с пакетом *Pillow*. Были освоены функции рисования фигур и линий.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from PIL import ImageDraw, Image


def triangle(img, x0, y0, x1, y1, thickness, color):
    fill_color=None):
    draw = ImageDraw.Draw(img)
    coordinates = [(x0, y0), (x1, y1), (x2, y2)]

    if fill_color:
          draw.polygon(coordinates, fill = tuple(fill_color), outline = 
tuple(color), width = thickness)
    else:
           draw.polygon(coordinates, outline = tuple(color), width = 
thickness)
    return img

def change_color(img, color=None):
    if img.size[0] == 0 or img.size[1] == 0 or color == None:
        return img
    colour = {}
    pixels = img.load()
    for pixel in img.getdata():
        if pixel in colour:
            colour[pixel] += 1
        else:
            colour[pixel] = 1
    sortcolour = sorted([(v,k) for (k,v) in colour.items()])
    res = [(v,k) for (k,v) in sortcolour[-1:]]
    for i in range(img.size[0]):
        for j in range(img.size[1]):
            if pixels[i,j]==res[0][0]:
                pixels[i,j]=tuple(color)
    return img
```

```
def collage(img, N, M):
    a = img.size
    img1 = img.crop((0, 0, a[0], a[1]))
    imgnew = Image.new("RGB", (a[0] * M, a[1] * N))
    for i in range(M):
        for j in range(N):
            imgnew.paste((img1), (i*a[0], j*a[1]))
    return imgnew
```