**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Информатика»**

Тема: Введение в архитектуру компьютера

|  |  |  |
| --- | --- | --- |
| Студент гр. 3341 |  | Че М.Б. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью данной лабораторной работы является изучение модуля PIL и практическое применение модуля на поставленной задачи.

## 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)

Функция должна создать коллаж изображений (это же изображение, повторяющееся NxM раз. (N раз по высоте, M раз по ширине) и вернуть его (новое изображение).

При необходимости можно писать дополнительные функции.

## Выполнение работы

Импортируем Image, ImageDraw из модуля PIL.

def tringle

Объявим переменную pic, с помощью которой будем добавлять треугольник на img. После если fill\_color имеет значение None, то рисуем треугольник без заливки, иначе с заливкой.

def change\_color

Объявим переменную colors, в которой будут храниться кортежи(цвета) заданного рисунка. С помощью переменных max\_num находим самый частый цвет, который будет храниться в color\_avg. После с помощью двойного цикла все пиксели рисунка, которые совпадают с самым встречающимся цветом, мы заменяем на заданные цвета в условии.

def\_collage

В переменные x y записываем ширину и высоту изображения, которое подавалось в аргументе функции, объявляем новую переменную(img\_paste), в которой будет новое храниться новое изображение, ширина которого равна x \* M, а высота y \* N. Создаем двойной цикл, в котором изображение img заполняет всю строку back M раз при помощи метода past, когда img заполнит всю строку первый цикл сместит back\_y на y пикселей, тем самым перейдя на новую строку и так N раз. В конце функция возвращает img\_paste.

## Тестирование

Результаты тестирования представлены в Таблице 1.

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1 |  |  | Задача 1 |
| 2 |  |  | Задача 2 |
| 3 |  |  | Задача 3 |

## Выводы

Выполнив данную лабораторную работу, я изучил модуль PIL и нашел ему практическое применение.

# Приложение А Исходный код программы

Исходный файл: main.py

import PIL

from PIL import Image, ImageDraw

def triangle(img, x0, y0, x1, y1, x2, y2, thickness, color, fill\_color):

pic = ImageDraw.Draw(img)

if fill\_color == None:

pic.polygon([(x0,y0), (x1, y1), (x2,y2)], None, (color[0], color[1], color[2]), thickness)

return img

else:

pic.polygon([(x0,y0), (x1, y1), (x2,y2)], (fill\_color[0], fill\_color[1], fill\_color[2]), (color[0], color[1], color[2]), thickness)

return img

# Задача 2

def change\_color(img, color):

colors = img.getcolors(10000)

max\_num = 0

color\_avg = (0,0,0)

for i in colors:

if max\_num < i[0]:

max\_num = i[0]

color\_avg = i[1]

img\_pix = img.load()

for x in range(img.size[0]):

for y in range(img.size[1]):

if img\_pix[x, y] == color\_avg:

img\_pix[x, y] = (color[0], color[1], color[2])

return img

# Задача 3

def collage(img, N, M):

x = img.size[0]

y = img.size[1]

img\_paste = Image.new("RGB", (x \* M, y \* N), (0, 0, 0))

for y\_paste in range(0, img\_paste.size[1], y):

for x\_paste in range(0, img\_paste.size[0], x):

img\_paste.paste(img, (x\_paste, y\_paste))

return img\_paste