**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. 3341 |  | Анисимов Д.А |
| Преподаватель |  | Глазунов С.А. |

Санкт-Петербург

2023

## Цель работы

Целью данной лабораторной работы является изучение модуля PIL языка python и его использование для решения задач, представленных в работе.

## 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 библиотеку Pillow.

1. Функция *triangle* принимает изображение *img* и координаты трех точек (*x0, y0, x1, y1, x2, y2*), которые образуют треугольник. Она также принимает толщину линии *thickness*, цвет линии *color* и цвет заливки *fill\_color*. Функция создает новую копию изображения и рисует на ней треугольник с заданными параметрами. Если *fill\_color* равен *None*, то треугольник будет только обведен цветом *color*, иначе он будет также заполнен цветом *fill\_color*. Возвращается новое изображение.

2. Функция *change\_color* принимает изображение *img* и новый цвет *color*. Она находит наиболее часто встречающийся цвет на изображении и заменяет его на новый цвет. Для этого функция сначала определяет все цвета на изображении и сортирует их по частоте встречаемости. Затем она копирует изображение, получает доступ к пикселям и заменяет цвета, соответствующие наиболее часто встречающемуся цвету, на новый цвет. Возвращается новое изображение.

3. Функция *collage* принимает изображение *img* и целочисленные значения *n* и *m*. Она создает новое изображение с размерами, заданными умножением ширины и высоты исходного изображения на *m* и *n* соответственно. Затем функция добавляет *n* раз по горизонтали и *m* раз по вертикали исходное изображение к новому изображению. Возвращается новое изображение-коллаж.

Этот код позволяет использовать функции для создания треугольников на изображении с заданными параметрами, изменения цвета наиболее часто встречающегося цвета на изображении и создания коллажей из одного и того же изображения.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в Таблице 1.

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1 | triangle(img, 125, 125, 0, 150, 0, 0, 1, (125, 125, 125), (0, 125, 0)) |  | Задача 1 |
| 2 | change\_color(Image.new("RGB", (350, 350), (255, 0, 0)), (0,0,255)) |  | Задача 2 |
| 3 | collage(img, 3, 3) |  | Задача 3 |

## Выводы

В результате проведения эксперимента были изучены и применены в практике функции библиотеки PIL языка Python. Кроме того, была разработана программа, которая эффективно решает три подзадачи.

# Приложение А Исходный код программы

Название файла: main.py

from PIL import Image

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

new\_img = img.copy()

draw = ImageDraw.Draw(new\_img)

if fill\_color is None:

draw.polygon([(x0,y0), (x1, y1), (x2,y2)], outline=(color[0], color[1], color[2]), width=thickness)

else:

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

return new\_img

def change\_color(img, color):

colors = img.getcolors(100000)

colors.sort(reverse=True, key=lambda x: x[0])

most\_common\_color = colors[0][1]

new\_img = img.copy()

pixels = new\_img.load()

width, height = new\_img.size

for x in range(width):

for y in range(height):

if pixels[x, y] == most\_common\_color:

pixels[x, y] = tuple(color)

return new\_img

def collage(img, n, m):

width, height = img.size

collage\_width = width \* m

collage\_height = height \* n

collage\_img = Image.new('RGB', (collage\_width, collage\_height))

for i in range(n):

for j in range(m):

collage\_img.paste(img, (j \* width, i \* height))

return collage\_img