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. 3342 _____ Иванов С.С.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2023

Цель работы

Целью работы является освоение работы с функциями и функционалом библиотеки Pillow.

Задание

Вариант 3.

Задача 1.

Рисование пентаграммы в круге

Необходимо написать функцию solve(), которая рисует на изображении пентаграмму в окружности.

Функция solve() принимает на вход:

- Изображение (img)
- Координаты центра окружности (x,y)
- Радиус окружности
- Толщину линий окружности (thickness)
- Цвет линий и окружности (color) – представляет собой список (list) из 3-х целых чисел

Функция должна изменить исходное изображение и вернуть его изображение.

Задача 2.

Поменять местами участки изображения и поворот

Необходимо реализовать функцию solve, которая меняет местами два квадратных, одинаковых по размеру, участка изображений и поворачивает эти участки на 90 градусов по часовой стрелке, а затем поворачивает изображение на 90 градусов по часовой стрелке.

Функция solve() принимает на вход:

- Квадратное изображение (img)
- Координаты левого верхнего угла первого квадратного участка(x0,y0)
- Координаты левого верхнего угла второго квадратного участка(x1,y1)
- Длину стороны квадратных участков (width)

Функция должна сначала поменять местами переданные участки изображений. Затем повернуть каждый участок на 90 градусов по часовой стрелке. Затем повернуть всё изображение на 90 градусов по часовой стрелке.

Функция должна вернуть обработанное изображение, не изменяя исходное.

Задача 3.

Средний цвет

Необходимо реализовать функцию solve, которая заменяет цвет каждого пикселя в области на средний цвет пикселей вокруг (не считая сам этот пиксель).

Функция solve() принимает на вход:

- Изображение (img)
- Координаты левого верхнего угла области (x0,y0)
- Координаты правого нижнего угла области (x1,y1)

Функция должна заменить цвета каждого пикселя в этой области на средний цвет пикселей вокруг.

Пиксели вокруг :

- 8 самых близких пикселей, если пиксель находится в центре изображения
- 5 самых близких пикселей, если пиксель находится у стенки
- 3 самых близких пикселя, если пиксель находится в угле

Выполнение работы

Написанная программа написана на языке Python с использованием библиотеки Pillow. Она состоит из 3-функций, которые вызываются сразу на сайте https://e.moevm.info.

Функция pentagram принимает на вход изображение (img), центральные координаты (x, y), радиус (r), толщину линии и цвет. Она рисует на изображении пятиконечную звезду на основе указанных параметров. Сначала она создает экземпляр класса ImageDraw. Затем рисует эллипс с указанным радиусом и цветом в центральной точке. После этого она вычисляет координаты пяти вершин пятиконечной звезды и рисует линии между ними, чтобы завершить форму. Функция возвращает измененное изображение.

Функция swap принимает на вход изображение (img), координаты двух квадратных областей (x0, y0 и x1, y1) и ширину квадратов. Она создает новое изображение и поворачивает две квадратные области на 90 градусов. Затем она меняет местами две повернутые области в новом изображении, создавая впечатление их обмена. Наконец, функция поворачивает всё новое изображение на 90 градусов и возвращает измененное изображение.

Функция avg_color принимает на вход изображение (img), координаты двух точек (x0, y0 и x1, y1) и вычисляет средний цвет в указанной области. Она создает копию входного изображения, затем перебирает пиксели в указанной области и вычисляет средний цвет на основе цветов окружающих пикселей. После обновления цвета каждого пикселя в пределах области она возвращает измененное изображение.

Разработанный программный код см. в приложении А.

Выводы

В результате выполнения лабораторной работы были получены ценные навыки работы с функциями и библиотркой Pillow, и они же применены на практике.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from numpy import (
    pi,
    sin,
    cos
)

from PIL import ImageDraw


def get_peak(get_fi, get_node, node_num):
    fi = get_fi(node_num)
    return get_node(fi)


def pentagram(img, x, y, r, thickness, color):
    draw = ImageDraw.Draw(img)

    draw.ellipse(
        [x - r, y - r, x + r, y + r],
        None,
        tuple(color),
        thickness
    )

    peaks = [
        get_peak(
            lambda n: (pi / 5) * (2 * n + 3 / 2),
            lambda fi: (int(x + r * cos(fi)), int(y + r * sin(fi))),
            n
        ) for n in range(5)
    ]

    for i in range(5):
        draw.line((peaks[i], peaks[(i + 2) % 5]), tuple(color), thickness)

    return img


def swap(img, x0, y0, x1, y1, width):
    new_img = img.copy()
    fi = -90

    picture1 = img.crop((x0, y0, x0 + width, y0 + width)).rotate(fi)
    picture2 = img.crop((x1, y1, x1 + width, y1 + width)).rotate(fi)

    new_img.paste(picture1, (x1, y1))
    new_img.paste(picture2, (x0, y0))

    new_img = new_img.rotate(fi)
```

```
    return new_img


def check_boundary(pixel, scope):
    return pixel[0] >= 0 and pixel[0] < scope[0] and pixel[1] >= 0 and
pixel[1] < scope[1]


def avg_color(img, x0, y0, x1, y1):
    new_img = img.copy()
    img_arr = new_img.load()
    shape_img = new_img.size

    for x in range(x0, x1 + 1):
        for y in range(y0, y1 + 1):

            near_pixels_cord = tuple(p for p in (
                    (x - 1, y - 1), (x, y - 1), (x + 1, y - 1), (x + 1,
y),
                    (x + 1, y + 1), (x, y + 1), (x - 1, y + 1), (x - 1, y)
                ) if check_boundary(p, shape_img)
            )

            near_pixels = tuple(img.getpixel(p) for p in near_pixels_cord)
            count_of_near_pixels = len(near_pixels)

            r = (p[0] for p in near_pixels)
            g = (p[1] for p in near_pixels)
            b = (p[2] for p in near_pixels)

            color = (
                int(sum(r) / count_of_near_pixels),
                int(sum(g) / count_of_near_pixels),
                int(sum(b) / count_of_near_pixels)
            )

            img_arr[x, y] = color

    return new_img
```