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 _____ Лапшов К.Н.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2023

Цель работы

Целью работы является освоение работы с функциями и библиотекой 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 получает на вход изображение, координаты центра окружности, радиус окружности, цвет окружности и линий, а также их толщину. Рисуется изображение при помощи метода “Image.Draw”. На созданной картинке, рисуется пентаграмма по таким параметрам как: координаты, цвет отрезка, его толщина.

Функция swap получает на вход изображение и координаты левого верхнего угла первого квадратного участка и правого верхнего угла второго квадратного участка. Данная функция, при помощи метода copy копирует рисунок. После этого с помощью метода crop вырезаются необходимые углы и поворачиваются на 90 градусов. Сразу после, поворачивается и вся картинка, результат возвращается из функции.

Функция avg_color принимает изображение и координаты прямоугольной области на изображении. Затем она вычисляет средний цвет пикселей в этой области, заменяя каждый пиксель на изображении средним цветом его окружения. Функция проходит через каждый пиксель в указанной области, вычисляет средний цвет по соседним пикселям и заменяет текущий пиксель этим цветом. Результат сохраняется в новом изображении, которое затем возвращается из функции.

Лямбда-функция check принимает координаты (x, y) и проверяет, находятся ли они в пределах размеров изображения img_shape. Если обе координаты x и y больше или равны 0 и меньше, чем соответствующие ширина и высота изображения, то функция возвращает True, в противном случае - False.

Данная программа демонстрирует использование функций библиотеки Pillow и работу функций на языке Python для выполнения различных графических операций

Разработанный программный код см. в приложении А.

Выводы

Была освоена библиотека Pillow. Полученные знания были применены на практике. Были разработаны такие функции как: рисование пентаграммы в круге, преобразование изображения путём перестановки участков изображения, нахождения среднего цвета пикселя.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from PIL import ImageDraw
import numpy as np


def pentagram(img, x, y, r, thickness, color):
    draw = ImageDraw.Draw(img)
    xy = [x - r, y - r, x + r, y + r]
    draw.ellipse(xy=xy, fill=None, outline=tuple(color), width=thickness)
    vertices = []
    for i in range(5):
        phi = (np.pi / 5) * (2 * i + 3 / 2)
        node_i = (int(x + r * np.cos(phi)), int(y + r * np.sin(phi)))
        vertices.append(node_i)
    for i in range(5):
          draw.line((vertices[i], vertices[(i + 2) % 5]), tuple(color),
thickness)
    return img


def swap(img, x0, y0, x1, y1, width):
    img_copy = img.copy()

    piece_one = img.crop((x0, y0, x0 + width, y0 + width)).rotate(-90)
    piece_two = img.crop((x1, y1, x1 + width, y1 + width)).rotate(-90)

    img_copy.paste(piece_one, (x1, y1))
    img_copy.paste(piece_two, (x0, y0))

    img_copy = img_copy.rotate(-90)

    return img_copy


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

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

            pix_ind = (
                 (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)
            )

             check = lambda xy: ((xy[0] >= 0) and (xy[0] < img_shape[0])
and (xy[1] >= 0) and (xy[1] < img_shape[1]))
```

```
            pix_ind = tuple(filter(check, pix_ind))
                  pix = tuple(map(img.getpixel, ((i[0], i[1]) for i in
pix_ind)))

            R, G, B = 0, 0, 0

            for i in range(len(pix)):
                R += pix[i][0]
                G += pix[i][1]
                B += pix[i][2]
                 res_color = tuple((R // len(pix), G // len(pix), B //
len(pix)))
            img_arr[x, y] = res_color

    return img_result
```