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 _____ Львов А.В.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург
2023

Цель работы

Целью работы является освоение работы с модулем Pillow языка Python.

Задание

Вариант 2

Предстоит решить 3 подзадачи, используя библиотеку Pillow (PIL).

1) Рисование пентаграммы в круге

Необходимо написать функцию pentagram(), которая рисует на изображении пентаграмму в круге.

Функция pentagram() принимает на вход:

- Изображение (img)
- координаты левого верхнего и правого нижнего угла квадрата, в который вписана окружность (x0, y0, x1, y1)
- Толщину линий и окружности (thickness)
- Цвет линий и окружности (color) представляет собой список (list) из 3-х целых чисел

Функция должна вернуть обработанное изображение.

2) Инвертирование полос

Необходимо реализовать функцию invert, которая делит изображение на "полосы" и инвертирует цвет нечетных полос (счёт с нуля).

Функция invert() принимает на вход:

- Изображение (img)
- Ширину полос в пикселах (N)
- Признак того, вертикальные или горизонтальные полосы (vertical - если True, то вертикальные)

Функция должна разделить изображение на вертикальные или горизонтальные полосы шириной N пикселей. И инвертировать цвет в нечетных полосах (счет с нуля). Последняя полоса может быть меньшей ширины, чем N.

3) Поменять местами 9 частей изображения.

Необходимо реализовать функцию mix, которая делит квадратное изображение на 9 равных частей (сторона изображения делится на 3), и по правилам, записанным в словаре, меняет их местами.

Функция mix() принимает на вход:

- Изображение (img)
- Словарь с описанием того, какие части на какие менять (rules)

Выполнение работы

Программа разработана на языке Python с использованием модуля Pillow.

Функция pentagram принимает на вход изображение, координаты левого верхнего и правого нижнего угла квадрата, в который вписана окружность (x0, y0, x1, y1), толщину линий и окружности (thickness), цвет линий и окружности (color) представляет собой список (list) из 3-х целых чисел. Она вычисляет координаты центра окружности, её радиус и при помощи цикла for записывает в список points координаты вершин пентаграммы при помощи модуля numpy (sin, cos, pi). В следующем цикле for она соединяет вершины при помощи метода line. Функция возвращает обработанное изображение.

Функция invert принимает на вход изображение (img), ширину полос в пикселах (N), признак того, вертикальные или горизонтальные полосы (vertical - если True, то вертикальные). Переменным width и height присваиваются значения ширины и высоты изображения соответственно. Далее, если vertical равен True, при помощи цикла for и методов crop, invert и paste происходит вырезание части изображения part, смена цвета и вставка этой части в исходное изображение соответственно. Функция возвращает обработанное изображение.

Функция mix принимает на вход изображение (img), словарь с описанием того, какие части на какие менять (rules). Переменным block_x и block_y присваиваются значения целочисленного деления ширины и длины на 3 соответственно, так как необходимо разделить изображение на 9 ровных частей. Далее, при помощи двух вложенных циклов for в списки parts и points добавляются вырезанные методом crop части изображения и их координаты соответственно. Затем, в цикле for при помощи метода paste вырезанные части изображения вставляются на новое место в соответствии с полученным словарём rules. Функция возвращает обработанное изображение.

Разработанный программный код см. в Приложении А.

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	img.size=(300, 300) ; x0 = 96; y0 = 136; x1 = 239; y1 = 279; thickness = 6; color = [38, 99, 226]; Функция – pentagram.	Корректный вывод
2.	img.size=(300, 300) ;N = 37; vertical = True; Функция – invert.	Корректный вывод
3.	img.size=(300, 300) ; rules = {0: 3, 1: 0, 2: 7, 3: 2, 4: 7, 5: 4, 6: 1, 7: 4, 8: 8} Функция mix.	Корректный вывод

Выводы

В ходе выполнения лабораторной работы была разработана программа на языке Python с использованием модуля Pillow, были освоены такие его методы, как crop, invert, paste и т.д.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from PIL import ImageDraw, ImageOps
from numpy import pi, cos, sin


def pentagram(img, x0, y0, x1, y1, thickness, color):
    rgb = tuple(color)
    drawing = ImageDraw.Draw(img)
    drawing.ellipse([(x0,   y0),   (x1,   y1)],   width=thickness,
outline=rgb)
    center_x = (x1 + x0) // 2
    center_y = (y1 + y0) // 2
    radius = (x1 - x0) // 2
    points = []
    for i in range(5):
        phi = pi * (2 * i + 3 / 2) / 5
        node_i = (int(center_x + radius * cos(phi)), int(center_y
+ radius * sin(phi)))
        points.append(node_i)
    for i in range(5):
        if i == 3:
            drawing.line([points[i],     points[0]],     fill=rgb,
width=thickness)
        elif i == 4:
            drawing.line([points[i],     points[1]],     fill=rgb,
width=thickness)
        else:
            drawing.line([points[i],  points[i  +  2]],  fill=rgb,
width=thickness)
    return img



def invert(img, N, vertical):
    width, height = img.size
    count = 0
    if vertical:
        for i in range(0, width, N):
            if count % 2:
                part = img.crop((i, 0, i + N, height))
                inverted = ImageOps.invert(part)
                img.paste(inverted, (i, 0))
            count += 1
    else:
        for i in range(0, height, N):
            if count % 2:
                part = img.crop((0, i, width, i + N))
                inverted = ImageOps.invert(part)
                img.paste(inverted, (0, i))
            count += 1

    return img
```

```
    def mix(img, rules):
        block_x = img.width // 3
        block_y = img.height // 3
        parts = []
        points = []
        for y in range(3):
            for x in range(3):
                part = img.crop((x * block_x, y * block_y, (x + 1) *
block_x, (y + 1) * block_y))
                points.append((x * block_x, y * block_y))
                parts.append(part)

        for i in range(9):
            img.paste(parts[rules[i]], points[i])

        return img
```