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. 3344		**Сьомак Д.А.**
Преподаватель		**Иванов Д.В.**

Санкт-Петербург

2023

Цель работы

Освоение работы с изображением на языке Python с помощью библиотеки Pillow.

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)

Выполнение работы

Задача 1:

Была объявлена функция triangle, которая принимает на вход изображение img, координаты вершин треугольника (x0, y0, x1, y1, x2, y2), толщину линии контура треугольника thickness, цвет линии контура color, цвет, которым внутри залит треугольник fill_color(если значение None, значит треугольник не залит). Был создан объект для рисования на исходном изображении draw (draw = ImageDraw.Draw(img)). Кортежи координат вершин были записаны в единый кортеж points. С помощью оператора if была осуществлена проверка наличия цвета заливки. В случае наличия цвета на исходном изображении рисуется треугольник, залитый внутри заданным цветом с помощью метода Draw.polygon, при отсутствии цвета рисуется треугольник без заливки с помощью того же метода Draw.polygon без указания параметра fill. Функция возвращает изменённое исходное изображение (с нанесённым на него треугольником).

Задача 2:

Была объявлена функция change_color, которая принимает на вход изображение img и заданный цвет color, на который нужно будет заменить самый часто встречающийся цвет. Объявляем словарь colors, получаем пиксельное представление картинки px с помощью метода load. При помощи двух вложенных циклов for пробегаем по всем пикселям исходного изображения при этом используем оператор if для заполнения словаря. С помощью colors.get(img.getpixel(coordinate), 0) != 0 проверяем наличие ключа в словаре (метод словаря get возвращает 0, если ключа нет и значение по данному ключу, если он есть; метод getpixel возвращает кортеж цвета пикселя по заданным координатам). В случае выполнения условия значение по заданному ключу (количество пикселей цвета, который задан в ключе) увеличивается на 1, при невыполнении - значение будет равно 1, так как оно встретилось впервые. Далее используется ещё один цикл for с оператором if

внутри для определения цвета, который встречается наибольшее количество раз. Цикл проходит по всем значениям словаря и проверяет каждое значение на равенство максимальному значению(max(colors.values()) находит максимальное значение среди прочих, метод словарей values возвращает все значения имеющихся ключей), при верном равенстве переменная наиболее частого цвета maxcolor принимает значение ключа, который принадлежит значению, удовлетворяющему равенству. Далее снова используем два вложенных цикла for и оператор if для замены цветов. Ещё раз проходим по всем пикселям и, если его цвет равен переменной maxcolor, заменяем цвет пикселя, удовлетворившего равенству(img.getpixel(coordinate) == maxcolor), на заданный во входных значениях функции. Функция возвращает изменённое изображение (изображение, где самый частый цвет заменён на заданный).

Задача 3:

Была объявлена функция collage, которая принимает на вход изображение img, количество изображений по оси Y - N и количество изображений по оси X - M. Объявляем переменные ширины и высоты width и height исходного изображения с помощью поля size. Объявляем переменные ширины и высоты коллажа collage_width и collage_height, равные высоте и ширине исходного изображения, умноженного на количество изображений на соответствующих осях. Далее формируем коллаж collage, как новое изображение (используем метод Image.new для создания изображения с параметрами нужными для коллажа collage_width, collage_height). Используем два вложенных цикла for для формирования координат, куда будут дорисовываться изображения, на коллаже collage. С помощью метода Image.paste располагаем исходное изображение на сформированных координатах (x * width, y * height) на коллаже. Функция возвращает новое изображение - коллаж, сформированный из исходного изображения.

Исходный код см. в приложении А

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	triangle(Image.new('RGB', (500,500),"black"), 250, 400, 100, 100, 400, 100, 7, (0,255,0), (250,0,0))	img	-
2.	change_color(Image.new('RGB',(400,400),"white"), (0,255,0))	img	-
3.	collage(Image.new('RGB', (300,300),"blue"),5,6)	collage	-

Выводы

Была освоена работа с изображением на языке Python. Была изучена библиотека Pillow, были получены практические навыки применения функций рисования, обработки и совмещения изображений.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Somak_Demid_lb2.py

```
import PIL

def triangle(img, x0, y0, x1, y1, x2, y2, thickness, color, fill_color):
    draw = ImageDraw.Draw(img)
    points = ((x0, y0), (x1, y1), (x2, y2))
    if fill_color is not None:
        draw.polygon(points,            fill=tuple(fill_color),outline=
tuple(color), width=thickness)
    else:
        draw.polygon(points, outline=tuple(color), width=thickness)
    return img

def change_color(img, color):
    colors = {}
    px = img.load()
    for x in range(img.width):
        for y in range(img.height):
            coordinate = x, y
            if colors.get(img.getpixel(coordinate), 0) != 0:
                colors[img.getpixel(coordinate)] += 1
            else:
                colors[img.getpixel(coordinate)] = 1

    for key, value in colors.items():
        if value == max(colors.values()):
            maxcolor = key

    for x in range(img.width):
        for y in range(img.height):
            coordinate = x, y
            if img.getpixel(coordinate) == maxcolor:
                px[coordinate] = tuple(color)
    return img

def collage(img, N, M):
    width, height = img.size
    collage_width = width * M
    collage_height = height * N
    collage = Image.new('RGB', (collage_width, collage_height))
    for y in range(N):
        for x in range(M):
            collage.paste(img, (x * width, y * height))
    return collage
```