

**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

Лодыгин И.А.

Преподаватель

Иванов Д.В.

Санкт-Петербург

2023

## **Цель работы**

Целью работы является освоение введения в архитектуру компьютера посредством библиотеки Pillow.

## Задание

1 вариант.

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.

## **Основные теоретические положения**

Модуль Pillow Python является библиотекой для обработки изображений. Он предоставляет широкий спектр функций для работы с изображениями, включая их открытие, сохранение, изменение размера, наложение фильтров, рисование и многое другое.

Основные функции модуля Pillow Python включают в себя:

1. **Открытие и сохранение изображений:** модуль позволяет открывать изображения в различных форматах (например, JPEG, PNG, GIF) и сохранять их в нужном формате.
2. **Изменение размера изображений:** с помощью модуля можно изменять размер изображений, как уменьшая, так и увеличивая их.
3. **Рисование на изображениях:** модуль позволяет рисовать на изображениях с помощью различных инструментов, таких как кисти, линии, текст и другие.
4. **Обработка цветов:** с помощью Pillow Python можно изменять цветовую гамму изображений, корректировать яркость, контрастность и другие параметры цвета.

Эти основные функции делают модуль Pillow Python мощным инструментом для работы с изображениями и обеспечивают его широкое применение в различных областях, таких как веб-разработка, компьютерное зрение, обработка фотографий и дизайн.

## Выполнение работы

Происходит импорт функций Image и ImageDraw из библиотеки Pillow:

```
from PIL import Image, ImageDraw
```

Происходит импорт библиотеки Numpy: `import numpy as np`

Функция `triangle` принимает изображение `img`, координаты вершин треугольника `x0, y0, x1, y1, x2, y2`, толщину линий `thickness`, цвет линий `color` и цвет заливки `fill_color`. Затем цвет преобразуется в кортеж с помощью функции `tuple`. Если `fill_color` не равен `None`, то он также преобразуется в кортеж. Создается объект `draw` с помощью метода `ImageDraw.Draw`, который позволяет рисовать на изображении. Затем создается список `points`, содержащий координаты вершин треугольника. С помощью метода `draw.polygon` изображение `img` заполняется треугольником с вершинами из списка `points`, указанным цветом `fill_color` для заливки и цветом `color` для контура, а также с заданной толщиной линии. На выходе из функции возвращается изображение `img` с нарисованным треугольником.

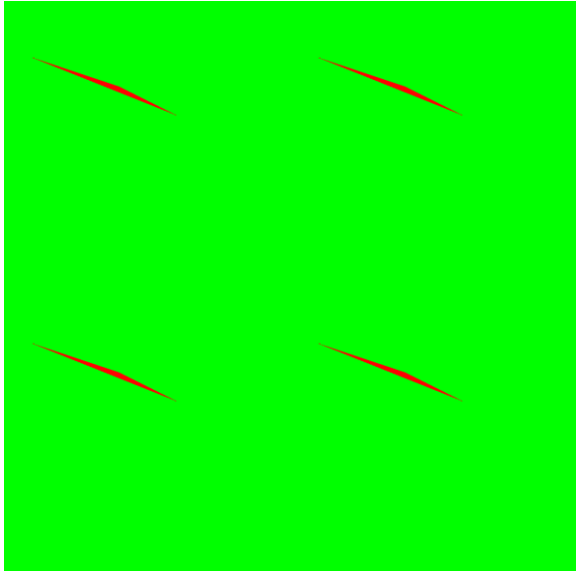
Функция `change_color` принимает два аргумента: `img` (изображение) и `color` (целевой цвет). Сначала изображение `img` преобразуется в массив `img_array` с помощью функции `np.array(img)`. Затем целевой цвет преобразуется в кортеж с помощью функции `tuple(color)` и сохраняется в переменной `target_color`. Далее находятся уникальные цвета и их количество в массиве `img_array` с помощью функции `np.unique()`, результаты сохраняются в переменных `colors` и `counts`. Наиболее часто встречающийся цвет определяется как цвет из массива `colors` с индексом, найденным с помощью функции `np.argmax(counts)`. Затем находятся индексы пикселей, которые имеют значение цвета, равное `most_common_color` с помощью функции `np.where((img_array == most_common_color).all(axis=-1))`. Найденные пиксели заменяются на целевой цвет `target_color` в массиве `img_array`. Наконец, создается новый объект изображения типа `Image` из массива `img_array` с помощью функции `Image.fromarray()` и возвращается.

Функция `collage` принимает в качестве аргументов `img` - исходное изображение, `N` - количество строк в коллаже и `M` - количество столбцов в коллаже. Затем она получает ширину и высоту исходного изображения с помощью метода `size`. Создается новое изображение `collage_img` с помощью метода `Image.new`, которое имеет формат 'RGB' и размер, равный умножению ширины исходного изображения на `M` и высоты на `N`. Это позволяет создать пустой холст для коллажа. Далее происходит вложенный цикл `for`, который проходит по каждой строке и столбцу в коллаже. Внутри цикла используется метод `paste` для вставки исходного изображения в `collage_img` в соответствии с текущими координатами (`j * width, i * height`), где `j` - номер столбца, а `i` - номер строки. После завершения циклов созданный коллаж `collage_img` возвращается из функции.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	<pre>img = Image.new('RGB', (500, 500), color = 'white') img = triangle(img, 50, 100, 200, 150, 300, 200, 2, (255, 0, 0), (255, 0, 0)) img = collage(img, 2, 2) img = change_color(img, (0, 255, 0)) img.show()</pre>	



## **Выводы**

Была освоена библиотека Pillow языка Python. Произошло ознакомление с архитектурой компьютера посредством освоения этой библиотеки.

Разработаны функции, которые, используя библиотеки Pillow и Numpy, позволяют делать обработку изображений, рисуя треугольник, создавая коллаж и заменяя наиболее часто встречающийся цвет.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: LR2CS.py

```
from PIL import Image, ImageDraw
import numpy as np

def triangle(img, x0, y0, x1, y1, x2, y2, thickness, color,
fill_color):
    color = tuple(color)
    if fill_color != None:
        fill_color = tuple(fill_color)
    draw = ImageDraw.Draw(img)
    points = [(x0, y0), (x1, y1), (x2, y2)]
    draw.polygon(points, fill = fill_color, outline = color,
width=thickness)
    return img

def change_color(img, color):
    img_array = np.array(img)
    target_color = tuple(color)
    colors, counts = np.unique(img_array.reshape(-1, 3), axis=0,
return_counts=True)
    most_common_color = colors[np.argmax(counts)]
    img_array[np.where((img_array == most_common_color).all(axis=-
1)))] = target_color
    return Image.fromarray(img_array)

def collage(img, N, M):
    width, height = img.size
    collage_img = Image.new('RGB', (M * width, N * height))
    for i in range(N):
        for j in range(M):
            collage_img.paste(img, (j * width, i * height))
    return collage_img
```