

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. 3343

Жучков О.Д.

Преподаватель

Иванов Д.В.

Санкт-Петербург

2023

Цель работы

Изучить библиотеку Pillow для работы с изображениями в Python и написать с помощью неё программу для обработки изображений.

Вариант 4

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 отрезка. Отрезок определяется:

- координатами начала
- координатами конца
- цветом
- толщиной.

Необходимо реализовать функцию `user_func()`, рисующую на картинке отрезок

Функция `user_func()` принимает на вход:

- изображение;
- координаты начала (`x0, y0`);
- координаты конца (`x1, y1`);
- цвет;
- толщину.

Функция должна вернуть обработанное изображение.

2) Преобразовать в Ч/Б изображение (любым простым способом).

Функционал определяется:

- Координатами левого верхнего угла области;
- Координатами правого нижнего угла области;
- Алгоритмом, если реализовано несколько алгоритмов преобразования изображения (по желанию студента).

Нужно реализовать 2 функции:

- `check_coords(image, x0, y0, x1, y1)` - проверяет координаты области (`x0, y0, x1, y1`) на корректность (они должны быть неотрицательными, не превышать размеров изображения, поскольку `x0, y0` - координаты

левого верхнего угла, x_1 , y_1 - координаты правого нижнего угла, то x_1 должен быть больше x_0 , а y_1 должен быть больше y_0);

- `set_black_white(image, x0, y0, x1, y1)` - преобразовывает заданную область изображения в черно-белый (используйте для конвертации параметр '1'). В этой функции должна вызываться функция проверки, и, если область некорректна, то должно быть возвращено исходное изображение без изменений. Примечание: поскольку черно-белый формат изображения (greyscale) является самостоятельным форматом, а не вариацией RGB-формата, для его получения необходимо использовать метод `Image.convert`.

3) Найти самый большой прямоугольник заданного цвета и перекрасить его в другой цвет. Функционал определяется:

- Цветом, прямоугольник которого надо найти
- Цветом, в который надо его перекрасить.

Написать функцию `find_rect_and_recolor(image, old_color, new_color)`, принимающую на вход изображение и кортежи rgb-компонент старого и нового цветов. Она выполняет задачу и возвращает изображение. При необходимости можно писать дополнительные функции.

Выполнение работы

Функция `user_func()` решает первую задачу, а именно: рисует линию по заданным координатам с заданным цветом и толщиной и возвращает получившееся изображение. Для рисования фигур на изображении используется интерфейс `ImageDraw.Draw`, в данной задаче метод `line()`.

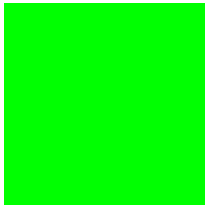

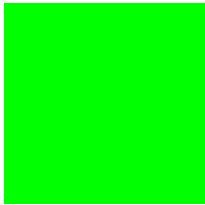
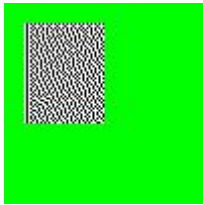
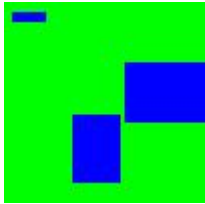
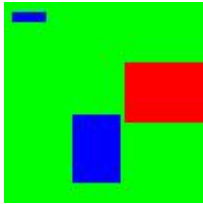
Вторая задача выполняется функцией `set_black_white()`. Перед конвертированием проверяются координаты функцией `check_coords()` (на случай выхода за границы), потом часть картинки вырезается с помощью `crop()` и конвертируется методом `convert()`, после чего вставляется обратно в исходное изображение методом `paste()`

Функция `find_rect_and_recolor()` заменяет прямоугольник определенного цвета и наибольшего размера, найденный функцией `find_biggest_rectangle()` на прямоугольник другого цвета с помощью `ImageDraw.Draw.rectangle()`.

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	 <code>user_func(img, 20, 30, 60, 80, (255,0,0), 5)</code>		Задача 1 выполняется верно
2.	 <code>set_black_white(img, 10, 10, 50, 60)</code>		Задача 2 выполняется верно
3.	 <code>find_rect_and_recolor(img, (0,0,255), (255,0,0))</code>		Задача 3 выполняется верно

Выводы

В ходе выполнения работы была изучена библиотека Pillow и написана программа для обработки изображений.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from PIL import Image, ImageDraw

# Задача 1
def user_func(image, x0, y0, x1, y1, fill, width):
    ImageDraw.Draw(image).line((x0, y0, x1, y1), fill, width)
    return image

# Задача 2
def check_coords(image, x0, y0, x1, y1):
    return all((0 <= x0 <= image.width, 0 <= x1 <= image.width,
                0 <= y0 <= image.height, 0 <= y1 <= image.height))

def set_black_white(image, x0, y0, x1, y1):
    if not check_coords(image, x0, y0, x1, y1):
        return image
    grayscale_area = image.crop((x0, y0, x1, y1)).convert("1")
    image.paste(grayscale_area, (x0, y0))
    return image

# Задача 3
def find_biggest_rectangle(image, color):
    rect_coords = (0,0,0,0)
    max_area = 1
    pixels = image.load()
    for x1 in range(image.width):
        for y1 in range(image.height):
            if pixels[x1, y1] != color:
                continue
            x2, y2 = x1, y1
            while x2 < image.width and pixels[x2,y1] == color:
                x2 += 1
            x2 -= 1
            while y2 < image.height and pixels[x1,y2] == color:
                y2 += 1
            y2 -= 1
            area = (x2 - x1 + 1) * (y2 - y1 + 1)
            if area > max_area:
                rect_coords = (x1, y1, x2, y2)
                max_area = area
    return rect_coords

def find_rect_and_recolor(image, old_color, new_color):
    rect_coords = find_biggest_rectangle(image, old_color)
    ImageDraw.Draw(image).rectangle(rect_coords, new_color)
    return image
```