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 _____ Бубякина Ю.В.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург
2023

Цель работы

Получение навыков обработки изображений с помощью языка Python.

Задание

Вариант 4

Предстоит решить 3 подзадачи, используя библиотеку Pillow (PIL). Для реализации требуемых функций студент должен использо-вать numpy и PIL. Аргумент image в функциях подразумевает объект ти-па <class 'PIL.Image.Image'>

1) Рисование отрезка. Отрезок определяется:

- координатами начала
- координатами конца
- цветом
- толщиной.

Необходимо реализовать функцию user_func(), рисующую на картинке отре-зок

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 ле-вого верхнего угла, x1, y1 - координаты правого нижнего угла, то x1 должен быть больше x0, а y1 должен быть больше y0);

- set_black_white(image, x0, y0, x1, y1) - преобразовывает заданную об-ласть изображения в черно-белый (используйте для конвертации пара-метр '1'). В этой функции должна вызываться функция проверки, и, ес-ли область некорректна, то должно быть возвращено исходное изоб-ражение без изменений. Примечание: поскольку черно-белый формат изображения (greyscale) является самостоятельным форматом, а не ва-риацией 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

Была импортирована библиотека PIL.

Была реализована функция user_func(image, x0, y0, x1, y1, fill, width), которая принимает на вход изображение(image), координаты начала отрезка(x0,y0), координаты конца отрезка(x1,y1), цвет(fill) и толщину(width). Для возможности рисовать на изображении в переменную drawing была записана функция ImageDraw.Draw(image). Далее к переменной drawing был применен метод drawing.line(((x0, y0, x1, y1)), fill, width), который рисует на изображении линию с заданными координатами, цветом и толщиной. Функция возвращает обработанное изображение.

В рамках 2 задачи были реализованы две функции: check_coords() и set_black_white(). Функция check_coords(image, x0, y0, x1, y1) принимает на вход изображение и координаты области, которую необходимо преобразовать в Ч/Б. Функция проверяет корректность заданных координат и возвращает True в случае успеха. Функция set_black_white(image, x0, y0, x1, y1) принимает на вход изображение и координаты области, которую необходимо преобразовать в Ч/Б. В случае, если вывод функции check_coords() оказался False, функция set_black_white() вернет исходное изображение. Если же проверяющая функция вывела True, то в переменную cut передается необходимая, вырезанная из исходного изображения, область. Далее в переменную cut_final с помощью метода cut.convert(‘1’) передается вырезанная ранее область в формате Ч/Б. Последним шагом является image.paste(cut_final, (x0, y0, x1, y1)), который вставляет в исходное изображение преобразованную область на место старой по тем же координатам. Функция возвращает обработанное изображение.

Была реализована функция find_rect_and_recolor(image, old_color, new_color), принимающая на вход изображение(image), цвет нужного прямоугольника(old_color) и цвет, на который его необходимо заменить(new_color) в RGB формате. Чтобы работать с изображением функция использует метод image.load() в переменной pixel_obj. В переменные width и height заносятся размеры изображения с помощью image.size. В переменных

max_area и max_coords будут храниться максимальная площадь прямоугольника и координаты левой верхней и правой нижней точек этого прямоугольника. Далее с помощью цикла for перебирается каждый пиксель и при совпадении его цвета с old_color с помощью цикла while начинается перебор пикселей, пока они нужного цвета, и формирование координат найденного прямоугольника. Когда найденный прямоугольник полностью «просканирован» в переменную area записывается его площадь по координатам x1 и y1. Далее эта площадь проверяется на максимальную и если на данный момент она наибольшая из найденных, то она записывается в переменную max_area, а в переменную max_coords записываются координаты левой верхней и правой нижней точек этого прямоугольника. Последний шаг – это двойной цикл for, который перебирает каждый пиксель на исходном изображении и при совпадении очередного пикселя с областью максимального прямоугольника с помощью метода image.putpixel((x, y), new_color) он перекрашивается в нужный цвет. Функция возвращает обработанное изображение.

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	user_func(Image.new(“RGB”, (251,251), “red”), 2,2,78,78,’yellow’, 5)	image	-
2.	check_coords(Image.new(“RGB”, (300,300), “lime”), 21,21,24,24)	True	-
3.	set_black_white(Image.new(“RGB”, (312,312), “red”), 34,34,123,123)	image	-
4.	find_rect_and_recolor(Image.new(“RGB”, (450,450), “black”), (255,255,255), (123,23,90))	image	-

Выводы

Освоены методы работы с библиотекой Pillow. Получены навыки работы с изображениями, рисования линий, смены формата изображения и изменение отдельных объектов изображения.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Bubyakina_Yuliya_lb2.py

```
from PIL import Image, ImageDraw

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

# Задача 2
def check_coords(image, x0, y0, x1, y1):
     if x0>=0 and y0>=0 and x1>=0 and y1>=0 and x0<=image.size[0]
and x1<=image.size[0] and y0<=image.size[1] and y1<=image.size[1] and
x0<=x1 and y0<=y1:
        return True
    return False

def set_black_white(image, x0, y0, x1, y1):
    if check_coords(image, x0, y0, x1, y1):
        cut = image.crop((x0,y0,x1,y1))
        cut_final = cut.convert('1')
        image.paste(cut_final,(x0,y0,x1,y1))
        return image
    return image

# Задача 3
def find_rect_and_recolor(image, old_color, new_color):
    old_color=tuple(old_color)
    new_color = tuple(new_color)
    pixel_obj = image.load()
    width,height = image.size
    max_area=0
    max_coords=()
    for x in range(width):
        for y in range(height):
            if pixel_obj[x,y]==old_color:
                area=1
                x1=1
                y1=1
                while x1+x<width and pixel_obj[x1+x,y]==old_color:
                    x1+=1
                while y1+y<height and pixel_obj[x,y1+y]==old_color:
                    y1+=1
                area=y1*x1
                if area>max_area:
                    max_area=area
                    max_coords=(x,y,x+x1-1,y+y1-1)
    for x in range (max_coords[0],max_coords[2]+1):
        for y in range(max_coords[1], max_coords[3] + 1):
            image.putpixel((x,y),new_color)
    return image
```