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 _____ Малиновский А.А.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург
2023

Цель работы

Изучение языка программирования Python и отработка его основных управляющих конструкций, таких как условия (if/elif/else), циклы (for/while), функции и операторы. Введение в архитектуру компьютера. Знакомство с библиотекой Pillow (PIL), работа с модулем numpy.

Задание (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, 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 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

Данный код представляет собой программу на языке программирования Python, которая решает несколько задач, связанных с решением задач на обработку изображений.

Представленный код начинается с импорта библиотеки Pillow (PIL) и модуля numpy. Библиотека Pillow дает возможность работы с изображениями (создание, обработка, преобразование и т.д.), эта библиотека будет использоваться в каждой из трех задач. Модуль numpy предоставляет функции для работы с массивами и матрицами, а также различные математические операции (он будет использован при решении одной из задач). Далее в коде объявляются пять функций (user_func(), check_coords(), set_black_white(), find_largest_rect(), find_rect_and_recolor()), каждая из которых участвует в решении какой-либо задачи. Рассмотрим данные функции.

Рассмотрим решение первой задачи. В нем используется всего одна функция user_func(). Функция user_func принимает на вход изображение image, координаты точек начала и конца линии (x0, y0) и (x1, y1), цвет линии fill и толщину линии width. Она использует модуль ImageDraw из библиотеки PIL для рисования линии на изображении. Сначала создается объект drawing, который позволяет рисовать на изображении. Затем drawing.line рисует линию на изображении. Затем функция возвращает измененное изображение.

Рассмотрим решение второй задачи. В нем используются функции check_coords() и set_black_white().

Функция check_coords принимает изображение image и координаты области (x0, y0) и (x1, y1). Она использует методы width и height объекта image, чтобы получить ширину и высоту изображения. Далее функция проверяет корректность введенных координат через сравнение их друг с другом и с высотой и шириной. Если данные корректны, функция возвращает True, иначе False.

Функция set_black_white принимает изображение image и и координаты области (x0, y0) и (x1, y1). Корректность введенных данных проверяется с помощью предыдущей функции. Если данные некорректны, возвращается исходное изображение, иначе функция продолжает работу. Далее из изображения вырезается указанная область и конвертируется в черно белый формат с помощью image.convert('1'). После этого вырезанный фрагмент вставляется на свое место в исходном изображении. Затем функция возвращает обработанное изображение.

Рассмотрим решение третьей задачи. В нем используются функции find_rect_and_recolor() и find_max_rectangle().

Функция find_largest_rect принимает изображение image и цвет color. Сначала функция преобразует изображение в массив array с помощью np.array(image). Далее заполняем 1 пиксели нужного нам цвета. После этого находим максимальную площадь и возвращаем координаты искомой области.

Функция find_rect_and_recolor принимает image old_color и new_color. Используя функцию find_largest_rect находим координаты наибольшего прямоугольника. В конце перекрашиваем прямоугольник в новый цвет и возвращаем изменённое изображение.

Разработанный программный код см. в приложении А. Результаты тестирования представлены в табл. 1, 2, 3.

Тестирование

Таблица 1 – Результаты тестирования первой функции

№п/п	Входные данные	Выходные данные	Комментарии
1.	image, x0, y0, x1, y1, fill, width	image	OK

Таблица 2 – Результаты тестирования второй функции

№п/п	Входные данные	Выходные данные	Комментарии
1.	image, x0, y0, x1, y1	image	OK

Таблица 3 – Результаты тестирования третьей функции

№п/п	Входные данные	Выходные данные	Комментарии
1.	image, old_color, new_color	image	OK

Выводы

В процессе выполнения работы я погрузился в изучение языка программирования Python. Я проработал на практике его основные конструкции. Я познакомился с библиотекой Pillow (PIL), и поработал с модулем numpy. Мною была разработана программа, решающая сразу три задачи через три отдельные функции.

В процессе написания программы (решения задачи) мною использовались: модуль numpy, библиотека Pillow, функции (def), условия (if), циклы (for).

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

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

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

# Задача 2
def check_coords(image, x0, y0, x1, y1):
    if  (x0 < x1 and y0 < y1 and x0 > 0 and y0 > 0
        and x1 < image.width and y1 < image.height):
        return True
def set_black_white(image, x0, y0, x1, y1):
    if check_coords(image,x0,y0,x1,y1):
        area=image.crop((x0,y0,x1,y1))
        converted_area=area.convert("1")
        image.paste(converted_area, (x0,y0,x1,y1))
    return image

# Задача 3
def find_largest_rect(image, color):
    # преобразование изображения в матрицу
    array=np.array(image).tolist()
    for i in range(len(array)):
        for j in range(len(array[i])):
            # устанавливаем значение 1, если пиксель имеет искомый цвет
            array [i][j]=int(array[i][j]==list(color))
    array=np.array(array)

    # считаем максимальную высоту
    for i in range (1,len(array)):
        for j in range(len(array[i])):
            if array[i][j]!=0:
                array[i][j]+=array[i-1][j]
    #ищем максимальную площадь
    max_area=0
    coordinates=(0,0,0,0)
    for i in range(len(array)):
        temp_area=0
        for k in set(array[i]):
            for j in range(len(array[i])):
                if k<=array[i][j]:
                    temp_area+=k
```

```
                if j==len(array[i])-1 or array[i][j+1]<k :
                    if max_area<temp_area:
                        max_area=temp_area
                        coordinates=(j-temp_area//k+1,i-k+1,j,i)
                    area=0
    return coordinates

def find_rect_and_recolor(image, old_color, new_color):
    coords=find_largest_rect(image,old_color)
    arr=np.array(image)
    arr[coords[1]:coords[3]+1, coords[0]:coords[2]+1,:3]=list(new_color)
    image =Image.fromarray(arr)
    return image
```