**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

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3341 |  | Игнатьев К.А. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью работы является изучение библиотеки Pillow, решение 3 подзадач с использованием библиотеки Pillow (PIL), работа с объектом типа <class 'PIL.Image.Image'>

## Задание

Вариант 4

Решить 3 подзадачи, используя библиотеку Pillow (PIL). Для реализации требуемых функций 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

1 задача:

Объявлена функция def user\_func(image, x0, y0, x1, y1,fill,width). Она принимает на вход изображение, координаты начала и конца отрезка, цвет отрезка и его толщину. С помощью метода Draw отрисовывается линия с использованием выше указанных этих параметров и возвращается полученное измененное изображение.

2 задача:

Объявляется функция check\_coords(image, x0, y0, x1, y1), которая проверяет корректность координат, с которыми будет производится работа. Если координаты меньше нуля, координаты больше или равны границ изображения и если координаты нижнего угла меньше координат верхнего угла, то возвращается значение False. В иных случаях возвращается значение True.

Объявляется функция set\_black\_white(image, x0, y0, x1, y1). Если координаты корректны (функция check\_coords вернула True), то исходное изображение обрезается, переводится в чёрно-белый формат и вставляется в начальное.

3 задача:

Объявляется функцию find\_max\_rect(pix, width, height, x,y,old\_color). Первоначальные значения координат левого верхнего угла записываются равными (0,0), нижнего правого – (максимальные координаты, являющиеся длинной изображения и высотой изображения соответственно), в массив с текущими проверяемыми пикселями сохраняются переданные значения координат (x,y). Далее, пока массив не пустой, из него берутся первые координаты: Если они находятся в пределах изображения и цвет данного пикселя равен старому цвету, тогда в массив с координатами левого верхнего угла записываются минимальные найденные координаты, с координатами правого нижнего – максимальные. Далее в массив с текущими проверяемыми координатами добавляются все смежные координаты для следующей проверки и цвет проверенного пикселя обращается в черный, чтобы избежать множественной проверки. Функция выводит кортеж с координатами левого верхнего угла и правого нижнего.

Объявляется функция find\_rect\_and\_recolor(image, old\_color, new\_color). Создается копия изображения для работы с ним и используется функция load() для перевода в пиксели. Определяется длинна и ширина изображения с помощью функции size. Программа проходит по пикселям копии и если находится пиксель старого цвета, в переменную coords сохраняется кортеж из координат левого верхнего и правого нижнего угла прямоугольника, найденных с помощью вспомогательной функции find\_max\_react. Далее определяется площадь найденного прямоугольника и если она оказывается больше площади максимальной – сохраняются координаты углов прямоугольника и максимальная площадь и приравнивается к найденной. Далее программа проходит по пикселям наибольшего прямоугольника в заданном изображении и меняем их цвет на новый. Функция возвращает измененное изображение.

См. приложение А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | image=Image.new("RGB", (500, 500), (0,0,0))  img=user\_func(image, 400, 400, 200, 200, (255,0,0,0), 3)  Image.\_show(img) |  | №1 |
|  | set\_black\_white(Image.open("input.png"), 250, 200, 700, 500) |  | №2 |
|  | image = Image.new("RGB", (350, 250), 'black')  image.paste(Image.new("RGB", (30,150), 'red'), (150, 50))  image.paste(Image.new("RGB", (40,140), 'red'), (200, 30))  find\_rect\_and\_recolor(image, (255, 0 , 0), (0, 0, 255)) |  | №3 |

## Выводы

Изучена библиотека Pillow, решены 3 подзадачи с использованием библиотеки Pillow (PIL).

Реализована программа, состоящая из трех задач, под каждую из которых выделена отдельная функция.

# ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

import PIL

from PIL import Image, ImageDraw

def user\_func(image, x0, y0, x1, y1, fill, width):

drawing = ImageDraw.Draw(image)

drawing.line((x0,y0,x1,y1),fill,width)

return image

def check\_coords(image, x0, y0, x1, y1):

width, height = image.size

if x0 < 0 or y0 < 0 or x1 < 0 or y1 < 0:

return False

if x0 >= width or x1 >= width or y0 >= height or y1 >= height:

return False

if x1 <= x0 or y1 <= y0:

return False

return True

def set\_black\_white(image, x0, y0, x1, y1):

if check\_coords(image, x0, y0, x1, y1):

region = image.crop((x0, y0, x1, y1))

region = region.convert('1')

image.paste(region, (x0, y0, x1, y1))

return image

def find\_max\_rect(pix, width, height, x,y,old\_color):

min\_cord = [0, 0]

max\_cord = [width, height]

curr = [(x, y)]

while len(curr) > 0:

x1, y1 = curr.pop()

if (0 <= x1 < width and 0 <= y1 < height and pix[x1, y1] == old\_color):

max\_cord = [min(max\_cord[0], x1), min(max\_cord[1], y1)]

min\_cord = [max(min\_cord[0], x1), max(min\_cord[1], y1)]

curr += [(x1 - 1, y1), (x1 + 1, y1), (x1, y1 - 1), (x1, y1 + 1)]

pix[x1, y1] = (0, 0, 0)

return (max\_cord[0], max\_cord[1], min\_cord[0], min\_cord[1])

def find\_rect\_and\_recolor(image, old\_color, new\_color):

img=image.copy()

pix=img.load()

width, height=img.size[0], image.size[1]

max\_rect=0

for x in range(width):

for y in range(height):

if pix[x,y]==old\_color:

cords=find\_max\_rect(pix, width, height, x,y,old\_color)

rect= (cords[3]+1 - cords[1])\*(cords[2]+1 - cords[0])

if rect>max\_rect:

max\_cords=cords

max\_rect=rect

res\_img=image.load()

for x in range (max\_cords[0], max\_cords[2]+1):

for y in range (max\_cords[1], max\_cords[3]+1):

res\_img[x,y]=new\_color

return image