**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 |  | Самокрутов А.Р. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью лабораторной работы является изучение модуля Pillow языка программирования Python. Для этого требуется решить три подзадачи с использованием библиотек Pillow и numpy. Необходимо разработать функции, которые работают с объектами типа *<class ‘PIL.Image.Image’>*.

## Задание

**1) Рисование пентаграммы в круге**

Необходимо написать функцию *pentagram()*, которая рисует на изображении пентаграмму в круге.

Функция *pentagram()* принимает на вход:

* Изображение (*img*)
* координаты левого верхнего и правого нижнего угла квадрата, в который вписана окружность (*x0, y0, x1, y1*)
* Толщину линий и окружности (*thickness*)
* Цвет линий и окружности (*color*) - представляет собой список (*list*) из 3-х целых чисел

Функция должна вернуть обработанное изображение.

Примечание:  
Вершины пентаграммы высчитывать по формуле:

*phi = (pi/5)\*(2\*i+3/2)*

*node\_i = (int(x0+r\*cos(phi)),int(y0+r\*sin(phi)))*

*x0, y0* - координаты центра окружности, в который вписана пентаграмма

*r* - радиус окружности

*i* - номер вершины от 0 до 4

**Подсказка: Округляйте все вычисляемые вами значения (кроме значений углов) до целых чисел.**

**2) Инвертирование полос**

Необходимо реализовать функцию *invert()*, которая делит изображение на "полосы" и инвертирует цвет нечетных полос (счёт с нуля).

Функция *invert()* принимает на вход:

* Изображение (*img*)
* Ширину полос в пикселах (*N*)
* Признак того, вертикальные или горизонтальные полосы (*vertical* - если *True*, то вертикальные)

Функция должна разделить изображение на вертикальные или горизонтальные полосы шириной *N* пикселей. И инвертировать цвет в нечетных полосах (счет с нуля).**Последняя полоса может быть меньшей ширины, чем *N*.**

**3) Поменять местами 9 частей изображения**

Необходимо реализовать функцию *mix()*, которая делит квадратное изображение на 9 равных частей**(сторона изображения делится на 3),** и  по правилам, записанным в словаре, меняет их местами.

Функция *mix()* принимает на вход:

* Изображение (*img*)
* Словарь с описанием того, какие части на какие менять (*rules*)

Пример словаря *rules*:

{0:1,1:2,2:4,3:4,4:5,5:3,6:8,7:8,8:8}

Элементы нумеруются слева-направо, сверху-вниз.

В данном случае нулевой элемент заменяется на первый, первый на второй, второй на четвертый, третий на четвертый и так далее.

Функция должна вернуть обработанное изображение.

Можно реализовывать дополнительные функции.

## Выполнение работы

Импортируется модуль *PIL*, из которого импортируются *Image*, *ImageDraw*, *ImageOps*, и модуль *numpy*.

Функция 1.

Объявляется функция *pentagram(img, x0, y0, x1, y1, thickness, color)*. Вычисляется радиус круга *radius*, а также *x* и *y* координаты его центра *x\_centrer* и *y\_center*. Создаётся пустой список *vertices* для хранения координат каждой из вершин фигуры в виде *2-tuple*. Далее с помощью цикла *for* обходятся все целые числа от 0 до 5 не включительно, и в каждой итерации цикла по формулам, описанным в условии задания, находятся координаты вершины, которые записываются в переменные *x\_vertex* и *y\_vertex* и добавляются в список *vertices*. По завершении цикла элементы списка vertices меняем местами таким образом, чтобы они описывали последовательность точек, которые необходимо соединить, чтобы получить пятиконечную звезду (изначально фигура была правильным пятиугольником). Далее на изображении рисуется эллипс с заданными параметрами и линия, соединяющая все вершины пентаграмма в нужном порядке с заданным цветом и толщиной линии. Функция возвращает получившееся изображение.

Функция 2.

Объявляется функция *invert(img, N, vertical)*. В переменные *width* и *height* записываются ширина и высота изображения, полученные с помощью метода *Image.size*. Если значение *vertical* равно *True*, то посредством цикла *for* производится обход по вертикальным полосам заданной ширины, они вырезаются из изображений с помощью метода *Image.crop*, их цвета инвертируются с помощью метода *ImageOps.invert*, после чего полосы вставляются через *Image.paste* на то место, из которого они были вырезаны. Аналогичные действия производятся при значении *vertical*, равном *False*, для горизонтальных полос. Функция возвращает полученное изображение.

Функция 3.

Объявляется функция *mix(img, rules)*. Аналогично второй функции находятся ширина и высота изображения. Создаётся пустой список *pieces*, в котором будут храниться части изображения. Посредством вложенного цикла *for* проходим по девяти (три на три) равным частям изображения. На каждой итерации цикла часть изображения вырезается методом *Image.crop*, после чего она и её координаты добавляются в список *pieces*. По завершении цикла проходим по всем ключам в словаре *rules* и вставляем с помощью метода *Image.paste* часть изображения в соответствующее место. Изображение и его координаты извлекаются из списка *pieces* по индексу, полученному по ключу словаря *rules*. Полученное изображение возвращается функцией.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | pentagram(Image.new('RGB', (400, 400), (0, 0, 0)), 100, 100, 200, 200, 3, (10, 52, 112)) |  | Первая функция |
|  | invert(Image.new('RGB', (400, 400), 'black'), 20, 0) |  | Вторая функция |
|  | mix(Image.open('example.jpg'), {0: 1, 1: 2, 2: 3, 3: 4, 4: 5, 5: 6, 6: 7, 7 : 8, 8: 0}) |  | Третья функция |

## Выводы

Были разработаны функции, которые работают с объектами типа *<class ‘PIL.Image.Image’>*, выполняющие соответствующие задачи, используя библиотек Pillow и numpy.

В результате выполнения лабораторной работы были изучены и использованы на практике функции библиотек Pillow и numpy.

# Приложение А Исходный код программы

Название файла: lab.py

import PIL

from PIL import Image, ImageDraw, ImageOps

import numpy as np

def pentagram(img, x0, y0, x1, y1, thickness, color):

img\_drawing = ImageDraw.Draw(img)

radius = abs(x0 - x1) // 2

x\_center = x0 + abs(x0 - x1) // 2

y\_center = y0 + abs(y0 - y1) // 2

vertices = []

for i in range(5):

angle = (np.pi / 5) \* (2 \* i + 3 / 2)

x\_vertex = int(x\_center + radius \* np.cos(angle))

y\_vertex = int(y\_center + radius \* np.sin(angle))

vertices.append((x\_vertex, y\_vertex))

vertices = [vertices[i] for i in [0, 2, 4, 1, 3, 0]]

img\_drawing.ellipse((x0, y0, x1, y1), outline=tuple(color), width=thickness)

img\_drawing.line(vertices, fill=tuple(color), width=thickness)

return img

def invert(img, N, vertical):

width, height = img.size

if vertical:

for pixel in range(1, width // N + 1, 2):

line\_coords = (pixel \* N, 0, (pixel + 1) \* N, height)

inverted\_line = img.crop(line\_coords)

inverted\_line = ImageOps.invert(inverted\_line)

img.paste(inverted\_line, line\_coords[:2])

else:

for pixel in range(1, height // N + 1, 2):

line\_coords = (0, pixel \* N, width, (pixel + 1) \* N)

inverted\_line = img.crop(line\_coords)

inverted\_line = ImageOps.invert(inverted\_line)

img.paste(inverted\_line, line\_coords[:2])

return img

def mix(img, rules):

width, height = img.size

pieces = []

for ver in range(3):

for hor in range(3):

piece\_width, piece\_height = width // 3, height // 3

crop\_coords = (hor \* piece\_height, ver \* piece\_width, (hor + 1) \* piece\_height, (ver + 1) \* piece\_width)

crop = img.crop(crop\_coords)

pieces.append([crop, (hor \* piece\_height, ver \* piece\_width)])

for i in rules.keys():

img.paste(pieces[rules[i]][0], pieces[i][1])

return img