**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. 3342 |  | Шушко Л.Д. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Цель работы – изучение работы модуля Pillow.

## Задание

Вариант 2

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 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.

## Выполнение работы

В коде программы используются модули PIL и numpy.

В функции pentagram используется метод Draw для рисования окружности и линий пентаграммы. Окружность рисуется по заданным параметрам, а для рисования линий находится радиус окружности и координаты ее центра, угол и точки начала и конца линий, и соединяединяются эти точки линиями. Функция возвращает измененное изображение.

В функции invert находятся размеры изображения, проверяется, если нужно разбить изображение на вертикальные полоски, то выполняется это деление и инвертируется цвет только нечетных полосок, иначе разбивается изображение на горизонтальные полоски и инвертируются только нечетные горизонтальные полоски. Функция возвращает измененное изображение.

В функции mix достаточно найти ширину 9 части изображения, так как оно квадратное, а затем изображение делится на 9 равных частей и в зависимости от значений словаря rules каждая часть меняется на другую. Функция возвращает измененное изображение.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| № п/п | Входные данные | Выходные данные |
|  | pentagram:  img.size=(300, 300) ; x0 = 56; y0 = 103; x1 = 169; y1 = 216; thickness = 2; color = [111, 30, 94] | Функция работает корректно |
|  | invert:  img.size=(300, 300) ;N = 100; vertical = True; | Функция работает корректно |
|  | mix:  img.size=(300, 300) ; rules = {0: 0, 1: 6, 2: 8, 3: 0, 4: 1, 5: 8, 6: 0, 7: 1, 8: 2} | Функция работает корректно |

## Выводы

Изучена работа модуля Pillow.

Разработана программа с использованием модуля Pillow, в которой использовались такие его методы, как Draw, crop, line, invert, paste, size.

# Приложение А Исходный код программы

Название файла: Shushko\_Leya\_lb2.py

from PIL import Image,ImageDraw, ImageOps

from numpy import pi, cos, sin

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

drawing = ImageDraw.Draw(img)

drawing.ellipse(((x0, y0), (x1, y1)), fill = None, outline = tuple(color), width = thickness)

r = abs(x1 - x0) // 2

center\_x = abs(x1 + x0) // 2

center\_y = abs(y1 + y0) // 2

points=[]

for i in range (5):

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

node\_i = (int(center\_x + r \* cos(phi)), int(center\_y + r \* sin(phi)))

points.append(node\_i)

for i in range (5):

if i == 3:

drawing.line((points[3], points[0]), fill = tuple(color), width = thickness, joint = None)

elif i == 4:

drawing.line((points[4], points[1]), fill = tuple(color), width = thickness, joint = None)

else:

drawing.line((points[i], points[i+2]), fill = tuple(color), width = thickness, joint = None)

return img

def invert(img, N, vertical):

width, height = img.size

index = 0

if vertical:

for i in range(0, width, N):

if index % 2 != 0:

strip = img.crop((i, 0, i + N, height))

strip = ImageOps.invert(strip)

img.paste(strip, (i, 0))

index += 1

else:

for i in range(0, height, N):

if index % 2 != 0:

strip = img.crop((0, i, width, i + N))

strip = ImageOps.invert(strip)

img.paste(strip, (0, i))

index += 1

return img

def mix(img, rules):

square\_width = img.width // 3

parts = []

points = []

for i in range(3):

for j in range(3):

part = img.crop((j \* square\_width, i \* square\_width, (j + 1) \* square\_width, (i + 1) \* square\_width))

parts.append(part)

points.append((j \* square\_width, i \* square\_width))

for x in range(9):

img.paste(parts[rules[x]], points[x])

return img