**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 |  | Поддубный В.А. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью работы являлось изучение и практическое применения принципов программирования на языке Python, при этом используя модуль *numpy*, в частности пакет *numpy.linalg*.

## Задание

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'>*

Задача 1. Содержательная постановка задачи

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 исходное обработанное изображение.

#### Задача 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) полосах (счет с нуля). Последняя полоса может быть меньшей ширины, чем N.

#### Задача 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

Мой программный код написан на языке Python и использует библиотеку Pillow для обработки изображений. Программа включает три функции, каждая из которых выполняет свою задачу.

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

Эта функция рисует пентаграмму на изображении img с центром в точке (x0 + (x1 - x0) / 2, y0 + (y1 - y0) / 2). Параметры x0 и y0 представляют собой координаты верхнего левого угла прямоугольника, описывающего пентаграмму, а x1 и y1 - координаты его нижнего правого угла. thickness задает толщину линий, а color - цвет в формате RGB.

Функция invert(img, N, vertical):

Эта функция инвертирует каждую N-ую линию изображения в зависимости от значения параметра vertical. Если vertical равен True, инвертируются вертикальные линии; в противном случае - горизонтальные. Параметр N определяет ширину (или высоту, в зависимости от vertical) каждой области, которая будет инвертирована.

Функция mix(img, rules):

Эта функция перемешивает изображение img, разбивая его на девять частей и изменяя порядок этих частей в соответствии с переданными правилами в виде списка rules. Каждый элемент списка rules представляет собой индекс для определенной части изображения. После перемешивания изображение возвращается.

Данный код демонстрирует использование библиотеки Pillow для работы с изображениями и реализации различных преобразований. Эти функции могут быть полезны при обработке изображений с целью создания графических эффектов или изменения их структуры в соответствии с заданными правилами.

Разработанный программный код см. в приложении А.

Тестирование см. в приложении Б.

## Выводы

Были изучены различные способы преобразования изображения, написаны функции с использованием библиотеки Pillow.

# Приложение А Исходный код программы

Название файла: main.py

from PIL import Image, ImageDraw, ImageOps

from numpy import pi,cos,sin

import math

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

color = tuple(color)

drawing = ImageDraw.Draw(img)

drawing.ellipse(((x0,y0),(x1,y1)),None,color,thickness)

r = round((x1-x0)/2)

nodes = {}

x = round(x0 + r)

y = round(y0 + r)

for i in range(5):

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

node = (int(x + r \* cos(phi)), int(y + r \* sin(phi)))

nodes[i]=[node,i+1]

if i==4:

nodes[i]=[node,0]

for i in range(5):

nextNode = nodes[nodes[i][1]]

nextNextNode = nodes[nextNode[1]]

drawing.line((nodes[i][0],nextNextNode[0]),color,thickness)

return img

def invert(img, N,vertical):

size = img.size

numOfLines = int(math.ceil(size[0]/N))

for i in range(numOfLines):

if vertical:

cropped = img.crop((N\*i,0,N\*(i+1),size[1]))

else:

cropped = img.crop((0,N\*i,size[1],N\*(i+1)))

if i%2!=0:

cropped = ImageOps.invert(cropped)

if vertical:

img.paste(cropped,(N\*i,0))

else:

img.paste(cropped,(0,N\*i))

return img

def mix(img, rules):

size = img.size

cropedSize = (size[0]//3,size[1]//3)

cropedImgs = []

for i in range(3):

for j in range(3):

left = (j\*cropedSize[0],i\*cropedSize[1])

right = ((j+1)\*cropedSize[0],(i+1)\*cropedSize[1])

cropedImg = img.crop((left[0],left[1],right[0],right[1]))

cropedImgs.append(cropedImg)

count=0

for i in range(3):

for j in range(3):

cropedImg = cropedImgs[rules[count]]

count+=1

img.paste(cropedImg,(j\*cropedSize[0],i\*cropedSize[1]))

return img

# Приложение Б тЕСТИРОВАНИЕ

Исходные изображения



Рисунок 1 – Изображение для функции invert (krab1.jpeg)



Рисунок 2 – Изображение для функции mix (krab1.jpeg)

Параметры функций:

1) pentagram

img = Image.new("RGB",(300,300),"blue")

x0=100  
y0=100  
x1=200  
y1=200  
thickness=4  
color=[255, 255, 161]

2) invert

img = Image.open("krab1.jpeg")

N=90  
vertical = True

3) mix

img=Image.open("krab1.jpeg")  
rules={0:2,1:2,2:2,3:5,4:5,5:5,6:8,7:8,8:8}

Результат:

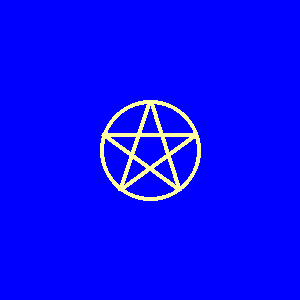


Рисунок 1 – Результат функции pentagram



Рисунок 2 – Результат функции invert



Рисунок 3 – Результат функции mix