**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. 3344 |  | Вердин К.К |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Освоение обработки изображений на языке Python.

## 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

Были импортированы библиотеки PIL, math.

Была создана функция def pentagram(img, x0, y0, x1, y1, thickness, color), приимающая на вход изображение img, координаты левого верхнего(x0, y0) и нижнего правого углов описанного прямоугольника (x1, y1), толщина линии thicknes и цвет линии color. Были инициализированы переменные radius, center\_x, center\_y для построения многоугольника определённого радиуса. При помощи цикла были найдены координаты вершин пентаграммы. Для отрисовки окружности на изображении был вызван *ImageDraw.Draw(img)*, у него был вызван метод *ellipse.* Для отрисовки линий пентаграммы использовался цикл *for*  и метод *line* у *ImageDraw.Draw(img):*

for i in range(0, 5):

coordinates\_line = (coordinates[i], coordinates[(i+2)%5])

img\_pentagram.line(coordinates\_line, color, thickness)

Функция возвращала отредактированное изображение.

В функции *def invert(img, N, vertical)*, принимающей на вход изображение, ширину полос для инвертирования в пикселях, признак расположения полос,было реализовано инвертирование цвета всех нечетных полос. Если признак положения являлся TRUE, то изображение поворачивалось на 270 градусов при помощи метода img.transpose(Image.Transpose.ROTATE\_270). Далее, при помощи цикла были получены координаты пикселей полосок цвет которых был инвертирован при помощиImageChops.invert(img.crop(box)). В конце если признак положения являлся TRUE изображение поворачивается ещё на 90 градусов. Функция возвращает изменённое изображение.

Была реализована функция def mix(img, rules), принимающая на вход изображение, словарь с описанием того, какие части на какие менять. Была инициализирована переменная side, которая была равна ширине пикселей одной части side = img.width // 3. Был создан массив кортежей parts вида ((x0, y0, x1, y1), где (x0, y0, x1, y1) – кортеж координат одной части. Далее, при помощи цикла for было создано изображение box по значению элемента parts[rules[i]]. При помощи метода ans.paste были заменены старые части на новые.

## Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | pentagram(Image.new("RGB", (300, 300), ‘black’), 80, 24, 253, 197, 5, [195, 11, 141]) | img | - |
|  | invert(Image.new("RGB", (300, 300), "black"), 33, False) | img | - |
|  | mix(Image.open("krab1.jpeg"), {0:2,1:2,2:2,3:5,4:5,5:5,6:8,7:8,8:8}) | img | - |

## Выводы

Была освоена обработка изображений на языке Python. Были получены базовые навыки работы с библиотекой *Pillow*. Были освоены функции рисования фигур и линий, отрисовки одного изображения на другое, отражения и поворота изображения, обрезки изображения.

# Приложение А Исходный код программы

Название файла: Verdin\_Kirill\_lb2.py

from PIL import Image, ImageDraw, ImageChops

import math

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

coordinates = []

color=tuple(color)

radius=abs(x1-x0)//2

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

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

for i in range(0, 5):

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

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

coordinates.append(node\_i)

img\_pentagram = ImageDraw.Draw(img)

img\_pentagram.ellipse(((x0, y0), (x1, y1)), None, color, thickness)

for i in range(0, 5):

coordinates\_line = (coordinates[i], coordinates[(i+2)%5])

img\_pentagram.line(coordinates\_line, color, thickness)

return img

def invert(img, N, vertical):

if vertical==True:

img=img.transpose(Image.Transpose.ROTATE\_270)

width=img.height

height=img.width

else:

width=img.width

height=img.height

for i in range(2, height//N + (height%N!=0) + 1, 2):

box = ((0, (i-1)\*N, width, i\*N))

part = ImageChops.invert(img.crop(box))

img.paste(part, box)

if vertical==True:

img=img.transpose(Image.Transpose.ROTATE\_90)

return img

def mix(img, rules):

ans=Image.new("RGB", (img.width, img.width), 'white')

side=img.width//3

parts=[]

for h in range(0, 3):

for w in range(0, 3):

parts.append(tuple((w\*side, h\*side, (w+1)\*side, (h+1)\*side)))

for i in range(0, 9):

box=img.crop(parts[rules[i]])

ans.paste(box, (parts[i][0], parts[i][1]))

return ans