**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 |  | Бажуков С.В. |
| 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

Вначале импортированы библиотеки Numpy и Pillow(PIL), а также необходимые классы из них.

Была реализована функция *def pentagram(img, x0, y0, x1, y1, thickness, color)*, принимающая на вход изображение - объект типа *<class 'PIL.Image.Image'>,* координаты левого верхнего и правого нижнего угла квадрата, в который вписана окружность, толщину линий и окружности, цвет линий и окружности, рисующая пентаграмму. Для отрисовки окружности на изображении был вызван *ImageDraw.Draw(img)*, у него был вызван метод *ellipse.* Далее были рассчитаны координаты центра окружности и ее радиус. Потом с помощью цикла *for*  были получены значения координат вершин пентаграммы и занесены в массив *cords.* Для отрисовки линий пентаграммы использовался цикл *for*  и метод *line* у *ImageDraw.Draw(img)*.

Затем была создана функция *pentagram(img, x0, y0, x1, y1, thickness, color),* которая принимает на вход изображение *img* , координаты левого верхнего и правого нижнего угла квадрата, в который вписана окружность, цвет линии и окружности. Эта функция рисует пентаграмму. Для этого вызван *ImageDraw.Draw(img)*, у него был вызван метод *ellipse, д*алее были рассчитаны координаты центра окружности и ее радиус. В цикле *for*  были получены значения координат вершин пентаграммы и занесены в массив *dots.* Для отрисовки линий пентаграммы использовался цикл *for*  и метод *line* у *ImageDraw.Draw(img)*.Функция возвращала изображение окружности с пентаграммой внутри.

Далее создана функция *invert(img, N, vertical)*, которая принимает на вход изображение, ширину полос для инвертирования в пикселях, признак расположения полос,было реализовано инвертирование цвета всех нечетных полос. Были получены ширина и высота изображения *width, height = img.size*. Далее, если признак расположения полос *vertical* являлся *True*, то выполнялся цикл *for*, который проходил по всем четным индексам частей ширины. В каждой итерации цикла создавался кортеж координат для части, которую надо инвертировать. С помощью метода *invert* у *ImageOps*, который принимал часть исходного изображения, создавалась инвертированная часть*.* Далее методом *paste* в исходное изображение вставлялась его инвертированная часть. Аналогично, если признак расположения полос не являлся *True*. Функция возвращала отредактированное изображение.

Затем была реализована функция *mix(img, rules)*, принимающая на вход изображение, словарь с описанием того, какие части на какие менять. Была инициализирована переменная *nw*, которая была равна ширине пикселей одной части.Затем были созданы 2 массива: *boxes -*  с координатами и *squares –* с частями изображения. Затем в двойном цикле *for* по очереди старые фрагменты изображения заменялись на новые в соответствии с порядком, указанном в словаре *rules.*

Функция возвращала отредактированное изображение.

## Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | pentagram(Image.new("RGB", (300, 300)), 45, 75, 85, 123, 5, [197, 114, 130]) | img | - |
|  | invert(Image.new("RGB", (300, 300), "black"), 19, False) | img | - |
|  | mix(Image.open("flower.jpeg"), {0:2,1:2,6:2,4:5,4:1,5:3,6:8,7:8,8:8}) | img | - |

## Выводы

В процессе выполнения работы была освоена обработка изображений на языке Python, получены базовые навыки работы с пакетом *Pillow*. Были освоены функции рисования фигур и линий, обработка изображений по заданным параметрам.

# Приложение А Исходный код программы

Название файла: src.py

from numpy import pi, sin, cos, ceil

from PIL import Image, ImageDraw, ImageOps

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

drawing = ImageDraw.Draw(img)

colour = tuple(color)

drawing.ellipse(((x0, y0), (x1, y1)), width=thickness, outline=colour)

r = (x1 - x0) // 2

x = (x1 + x0) // 2

y = (y1 + y0) // 2

dots = []

for i in range(5):

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

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

dots.append(node\_i)

for i in range(2):

crd = (dots[i-2], dots[i], dots[i+2])

drawing.line(crd, fill=colour, width=thickness)

drawing.line((dots[2], dots[-1]), fill=colour, width=thickness)

return img

def invert(img, N, vertical):

width, height = img.size

if vertical:

count = int(ceil(width / N))

for i in range(1, count, 2):

x1 = N\*i

x2 = N\*(i+1)

if x2 > width:

x2 = width

y1 = 0

y2 = height

img.paste(ImageOps.invert(img.crop((x1,y1,x2,y2))), (x1, y1))

else:

count = int(ceil(height / N))

for i in range (count):

if i%2!=0:

x1=0

x2 = width

y1 = N\*i

y2 = N\*(i+1)

if y2 > height:

y2 = height

img.paste(ImageOps.invert(img.crop((x1,y1,x2,y2))), (x1, y1))

return img

def mix(img, rules):

nw = img.width//3

squares=[]

boxes = []

for i in range(1, 4):

for j in range(1, 4):

square = img.crop(((j - 1) \* nw, (i - 1) \* nw, j \* nw,i \* nw))

squares.append(square)

boxes.append(((j - 1) \* nw, (i - 1) \* nw, j \* nw,i \* nw))

for orig, final in rules.items():

img.paste(squares[final], boxes[orig])

return img