**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 |  | Пачев Д.К. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

**Цель работы**

Написать программу на языке Python, выделив каждую задачу в отдельную функцию, а также освоить на практических задачах модуль Pillow.

**Задание**

Вариант 2.Вариант 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**

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

В переменной img\_draw создается объект ImageDraw и вызывается метод draw() для рисования. Радиус окружности r высчитывается по формуле (x1 – x0) // 2, также определяются координаты центра окружности circle\_x = (x0 + x1) // 2 и circle\_y = (y1 + y0) // 2. Далее с помощью метода ellipse() рисуется окружность, затем с помощью цикла в список pent\_coords добавляются координаты для построения пентаграммы. С использованием цикла for и метода line () рисуется сама пентаграмма. Возвращает функция обработанное изображение.

* Функция invert(img, N, vertical):

В переменной num\_of\_strip хранится число полос, на которые необходимо разделить фото. Циклом for проходимся по нечетным полосам и с помощью метода invert инвертируем их цвет, а методом paste() вставляем измененные полосы в изображение. Возвращает функция измененное изображение.

* Функция mix(img, rules):

С помощью цикла for заполняется данными словарь paste\_coords, в котором хранятся координаты квадратов, на которые делится изображение, а также в список img\_paste\_list добавляются копии этих квадратов, которые были получены методом crop(). Далее опять же циклом for и методом paste() изменяется исходное изображение в соответствии с указанными правилами rules. Возвращает функция обработанное изображение.

**Тестирование**

Результаты тестирования представлены в Таблице 1

Таблица 1 - Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1. | pentagram(Image.new(‘RGB’,(300,300),’red’),9,28,129,148,3,[150,178,182]) | img |  |
| 2. | invert(Image.new(‘RGB’,(300,300),’red’), 100, False) | img |  |
| 3. | 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**

В ходе выполнения лабораторной работы была разработана программа на языке Python, в которой каждая подзадача была выделена в отдельную функцию, а также изучены основные методы работы с Pillow.

# ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

from PIL import Image, ImageDraw, ImageChops  
from math import pi, sin, cos  
  
  
def pentagram(img, x0, y0, x1, y1, thickness, color):  
 r = (x1 - x0) // 2  
 img\_draw = ImageDraw.Draw(img)  
 color = tuple(color)  
 circle\_x = (x0 + x1) // 2  
 circle\_y = (y0 + y1) // 2  
 img\_draw.ellipse(((x0, y0), (x1, y1)), outline=color, width=thickness)  
 pent\_coords = []  
 for i in range(5):  
 phi = (pi / 5) \* (2 \* i + 3 / 2)  
 node\_i = (int(circle\_x + r \* cos(phi)), int(circle\_y + r \* sin(phi)))  
 pent\_coords.append(node\_i)  
 for i in range(2):  
 img\_draw.line([pent\_coords[i - 2], pent\_coords[i], pent\_coords[i + 2]], fill=color, width=thickness)  
 img\_draw.line([pent\_coords[2], pent\_coords[4]], fill=color, width=thickness)  
 return img  
  
  
def invert(img, N, vertical):  
 width, height = img.size  
 if vertical == True:  
 num\_of\_strip = width // N  
 for i in range(1, num\_of\_strip + 1, 2):  
 box = (N \* i, 0, N \* (i + 1), height)  
 invert\_img = ImageChops.invert(img.crop(box))  
 img.paste(invert\_img, box)  
 else:  
 num\_of\_strip = height // N  
 for i in range(1, num\_of\_strip + 1, 2):  
 box = (0, N \* i, width, N \* (i + 1))  
 invert\_img = ImageChops.invert(img.crop(box))  
 img.paste(invert\_img, box)  
 return img  
  
  
def mix(img, rules):  
 width, height = img.size  
 rect\_width = width // 3  
 rect\_height = height // 3  
 row\_count = 1;  
 paste\_coords = {}  
 img\_paste\_list = []  
 for i in range(9):  
 box = (rect\_width \* (i % 3), rect\_width \* (row\_count - 1), rect\_width \* ((i % 3) + 1), rect\_height \* row\_count)  
 if i != 0 and (i + 1) % 3 == 0:  
 row\_count += 1  
 paste\_coords[i] = box  
 img\_paste\_list.append(img.crop(box))  
 for i in range(9):  
 opt = rules[i]  
 img.paste(img\_paste\_list[opt], paste\_coords[i])  
 return img