**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. 2381 |  | Рыжиков И.С. |
| Преподаватель |  | Шевская Н.В. |

Санкт-Петербург

2022

## Цель работы

Изучить библиотеку Pillow. Применить полученные знания для решения задач, представленных в работе.

## 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.

Можно реализовывать дополнительные функции.

## Выполнение работы

Импортируем модуль numpy и пакеты библиотеки Pillow: Image, ImageDraw, ImageOps для работы c изображениями.

### 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.

Возвращает обработанное изображение.

Приводим список color к кортежу для работы с функциями пакета ImageDrow.

Вычисляем радиус и центр окружности, вписанной в квадрат.

Находим вершины пентаграммы points: вычисляем вершины пятиугольника только проходим их через одну.

Создаем объект draw для рисования.

Рисуем вписанную в квадрат окружность. Рисуем ломанную в виде звезды, с помощью метода line.

Возвращаем полученное изображение.

### Функция invert

Принимает на вход:

Изображение (img)

Ширину полос в пикселах (N)

Признак того, вертикальные или горизонтальные полосы (vertical - если True, то вертикальные)

Возвращает обработанное изображение.

Функция разделяет изображение на вертикальные или горизонтальные полосы шириной N пикселей. И инвертирует цвет в нечетных полосах (счет с нуля).

Инвертируем цвета изображения.

В зависимости от значения vertical в цикле for копируем в исходное изображение из инвертированного нечетные вертикальные (vertical == True) или горизонтальные (vertical == False) полосы. Для этого вычисляем значения начала и конца полосы, если конец полосы выходит за границы изображения, считаем его равным ширине/высоте.

Возвращаем полученное изображение.

### Функция pentagram

Принимает на вход:

* Изображение (img)
* Словарь с описанием того, какие части на какие менять (rules)

Возвращает обработанное изображение.

Функция делит квадратное изображение на 9 равных частей (сторона изображения делится на 3), и по правилам, записанным в словаре, меняет их местами.

Получаем размеры изображения, находим длину стороны одной части.

Создаем пустое новое изображение.

В цикле проходимся по номеру n каждому региону. Из словаря rules достаем номер m другого региона, на который требуется заменить. Копируем регион m в пустой регион n.

Возвращаем полученное 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 | Комментарии |
| Функция pentagram | | | |
|  |  |  | pentagram(img, 20, 30, 240, 250,  3, (200, 110, 50))  OK |
| Функция invert | | | |
|  |  |  | invert(img, 120, False)  OK |
|  |  |  | invert(img, 75, True)  OK |
| Функция pentagram | | | |
|  |  |  | mix(img, {0: 1, 1: 2, 2: 4, 3: 4, 4: 5, 5: 3, 6: 8, 7: 8, 8: 8})  OK |

## Выводы

Были изучена библиотека Pillow.

Разработаны функция pentagram, рисующая пентаграмму в заданном квадрате на изображении, функция invert, инвертирующая цвет чередующихся полос заданной ширины, функция mix, меняющая части изображения местами.

# Приложение А Исходный код программы

Название файла: main.py

*import* numpy *as* np

*from* PIL *import* Image, ImageDraw, ImageOps

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

    color = tuple(color)

*# Calculate*

    radius = int(abs(x1 - x0) / 2)

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

    center\_y = int(y1 - abs(x1 - x0) / 2)

    points = []

*for* i *in* range(6):

        phi = (np.pi/5)\*(4\*i+3/2)

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

                  int(center\_y+radius\*np.sin(phi)))

        points.append(node\_i)

*# Draw*

    draw = ImageDraw.Draw(img, "RGB")

    draw.ellipse((x0, y0, x1, y1), width=thickness, outline=color)

    draw.line(points, width=thickness, fill=color)

*return* img

def invert(img: Image.Image, N: int, vertical: bool):

    img\_invert = ImageOps.invert(img)

    width, height = img.size

*if* vertical:

*for* i *in* range(1, width // N+1, 2):

            start = i\*N

            end = (i+1)\*N *if* (i+1)\*N <= width *else* width

            region = img\_invert.crop((start, 0, end, height))

            img.paste(region, (start, 0, end, height))

*else*:

*for* i *in* range(1, height // N+1, 2):

            start = i\*N

            end = (i+1)\*N *if* (i+1)\*N <= height *else* height

            region = img\_invert.crop((0, start, width, end))

            img.paste(region, (0, start, width, end))

*return* img

def mix(img: Image.Image, rules: dict):

    width, height = img.size

    region\_length = width // 3

    new\_img = Image.new("RGB", (width, height), "black")

*for* n *in* range(9):

        m = rules[n]

        region = img.crop((m % 3 \* region\_length, m // 3 \* region\_length,

                          (m % 3 + 1) \* region\_length, (m // 3 + 1) \* region\_length))

        new\_img.paste(region, (n % 3 \* region\_length, n // 3 \* region\_length,

                      (n % 3 + 1) \* region\_length, (n // 3 + 1) \* region\_length))

*return* new\_img