

**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**

Студентка гр. 2381

Фомина А. К.

Преподаватель

Шевская Н.В.

Санкт-Петербург

2022

## **Цель работы.**

Изучить модуль Pillow и получить практические навыки, решив 3 подзадачи.

## **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

### **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)

### **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`)

### **Выполнение работы.**

Подключение модуля *numpy* с псевдонимом *np*. Также был подключен модуль *PIL*.

### **Функция `pentagram`**

На вход объявляются аргументы (`img, x0, y0, x1, y1, thickness, color`), по условию принимающие изображения, координаты, толщину линии и цвет. По формуле находится радиус, с помощью которого можно найти координаты центра окружности. Строится окружность с помощью метода `ellipse`. В цикле `for` высчитываются вершины пентаграммы. Также с помощью цикла строятся линии. Функция возвращает исходное изображение, на котором была нарисована пентаграмма.

### **Функция `invert`**

На вход объявляются три аргумента (`img, N, vertical`), по условию принимающие изображение, ширину полос и являются ли полосы вертикальными. Если аргумент *vertical* задается как `True`, то с помощью цикла `for` проходимся по ширине изображения. Проверяется является ли полоса нечётной и, если она нечётная, то с помощью метода `crop()` вырезается необходима область и инвертируется её цвет с помощью метода `invert()`. Затем вставляется в исходное изображение с помощью метода `paste()`. Аналогичное проводится с изображением, если аргумент *vertical* задается как `False`, но в данном случае проходимся по длине изображения. Функция возвращает исходное изображение, в которое были внесены изменения.

### **Функция `mix`**

На вход объявляются две переменные (`img, rules`), по условию принимающие изображение и словарь с описанием, какие части на какие надо менять. Создается новое изображение на основе поступившего. Также создается



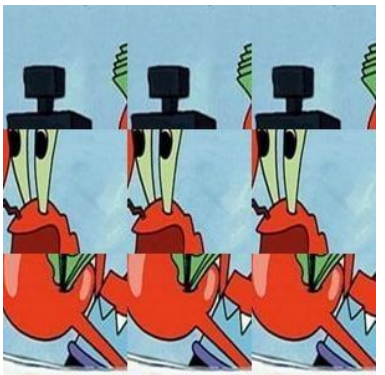
массив, в котором будут храниться координаты верхних углов квадратов, на которые будет разделено изображение. Далее по словарю из исходного изображения вырезаются квадраты с помощью метода *crop()*, из которых с помощью метода *paste()* собирается картинка. Функция возвращает полученное изображение.

Разработанный программный код см. в приложении А.

### Тестирование.

Результаты тестирования представлены в тбл.1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	Img, 200, 50, 250, 100, (0, 0, 0)		Верный ответ
2.	Img, 120, False		Верный ответ
3.	Img, {0 : 3, 1 : 3, 2 : 3, 3 : 5, 4 : 5, 5 : 5, 6 : 7, 7 : 7, 8 : 7}		Верный ответ

### **Вывод.**

Был изучен модуль Pillow, получены практические навыки. Были решены 3 подзадачи, с помощью написания функций **pentagram()**, **invert()**, **mix()**.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main\_lb2.py

```
import numpy as np
from PIL import Image, ImageDraw, ImageOps

def pentagram(img, x0, y0, x1, y1, thickness, color):
    r = abs(x0 - x1) // 2
    x = x0 + r
    y = y0 + r
    draw = ImageDraw.Draw(img, 'RGB')
    draw.ellipse((x0, y0, x1, y1), None, tuple(color), thickness)
    points = []
    for i in range(5):
        phi = (np.pi/5)*(2*i+3/2)
        node_i = (int(x+r*np.cos(phi)),int(y+r*np.sin(phi)))
        points.append(node_i)
    for i in range(5):
        draw.line((points[i], points[(i+2)%5]), tuple(color),
thickness)
    return img

def invert(img, N, vertical):
    if (vertical):
        for i in range(img.size[0]):
            if ((i // N) % 2 == 1):
                img_1 = ImageOps.invert(img.copy().crop((i, 0, i +
1, img.size[1])))
                img_2 = (i, 0)
                img.paste(img_1, img_2)
    else:
        for i in range(img.size[1]):
            if ((i // N) % 2 == 1):
                img_1 = ImageOps.invert(img.copy().crop((0, i,
img.size[0], i + 1)))
                img_2 = (0, i)
                img.paste(img_1, img_2)
    return img

def mix(img, rules):
    mix_img = Image.new("RGB", (img.size[0], img.size[1]), color=0)
    squares = []
    for i in range(3):
        for j in range(3):
            squares.append(
                (j * img.size[0] // 3, i * img.size[0] // 3, (j + 1)
* img.size[0] // 3, (i + 1) * img.size[0] // 3))
    for i in range(9):
        img_1 = img.copy().crop(squares[rules[i]])
        img_2 = (squares[i][0], squares[i][1])
        mix_img.paste(img_1, img_2)
    return mix_img
```