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. 3341 _____ Байрам Э.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2023

Цель работы

Работа направлена на разработку программного обеспечения, способного успешно выполнять предоставленные задачи с использованием языка Python и библиотеки Pillow для визуальной манипуляции. Этот код способен выполнять различные задачи, такие как рисование пентаграммы, инверсия цветов и перемещение частей изображения.

Задание

Рисование пентаграммы:

Используя библиотеку Pillow, нарисовать пентаграмму внутри круга.

Круг будет вписан в определенный квадрат.

Для рисунка будут использованы определенная толщина линий и цвет.

Инверсия цветов:

Изображение будет разделено на полосы определенной ширины.

Полосы могут быть вертикальными или горизонтальными, в зависимости от параметра vertical.

Цвет нечетных полос будет инвертирован.

Перестановка частей изображения:

Изображение будет разделено на девять равных частей (3x3 матрица).

В соответствии с определенным правилом, части будут меняться местами.

Правила замены будут представлены в виде словаря.

Выполнение работы

Нарисовать пентаграмму (pentagram функция):Эта функция, используя библиотеку Pillow, добавляет пентаграмму на изображение с определенными координатами и параметрами.

Окружность вписывается в квадрат с заданными координатами. Внутри нарисованного изображения содержится круг.

Углы пентаграммы вычисляются с использованием тригонометрии, а цвет и толщина задаются определенными параметрами.

Функция возвращает обработанное изображение.

Инверсия цветов (invert функция):

Эта функция разделяет изображение на полосы определенной ширины и инвертирует цвета в соответствии с заданными правилами.

Определяется, вертикальные или горизонтальные полосы используются, с использованием параметра vertical.

Цвета нечетных полос инвертируются, и функция возвращает обработанное изображение.

Перестановка частей изображения (mix функция):

Эта функция делит изображение на девять равных частей (3x3 матрица) и меняет их местами в соответствии с заданными правилами.

Правила изменения задаются в виде словаря. Каждая часть заменяется другой частью в соответствии с определенным правилом.

Функция возвращает обработанное изображение.

Эти задачи направлены на то, чтобы вы успешно освоили программирование на языке Python, использование библиотеки Pillow и базовые тригонометрические вычисления для выполнения различных визуальных манипуляций. Специально разработанные функции предназначены для успешного выполнения конкретных задач по обработке изображений.

Выводы

Эти задачи, успешное их выполнение, позволит мне улучшить навыки программирования на языке Python, более эффективно использовать внешние библиотеки. Этот опыт также укрепит мои базовые навыки в области обработки изображений, предоставив возможность разработки решений для реальных задач. Эффективное использование внешних библиотек, таких как Pillow, даст мне понимание того, как интегрировать подобные инструменты в различные проекты и приложения, что в свою очередь сделает мою работу более продуктивной. Улучшение навыков обработки изображений, особенно в контексте компьютерных наук, искусственного интеллекта или аналогичных областей, откроет возможность перейти к более сложным проектам и участвовать в продвинутых исследованиях. Благодаря подобным практическим заданиям я смогу расширить свои знания и опыт, что впоследствии позволит мне эффективно участвовать в более сложных проектах и задачах.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

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

def pentagram(img, x0, y0, x1, y1, thickness, color):
    drawing = ImageDraw.Draw(img)

    radius = int((x1 - x0) / 2)
    center_x = int((x1 + x0) / 2)
    center_y = int((y1 + y0) / 2)

    nodes = []
    for i in range(5):
        phi = (np.pi) * (2 * i + 3 / 2) / 5
        node_i = (int(center_x + radius * np.cos(phi)), int(center_y
+ radius * np.sin(phi)))
        nodes.append(node_i)

    nodes = [nodes[0], nodes[2], nodes[4], nodes[1], nodes[3],
nodes[0]]

    drawing.ellipse((x0, y0, x1, y1), outline=tuple(color),
width=thickness)
    drawing.line(nodes, fill=tuple(color), width=thickness)

    return img


def invert(img, N, vertical):
    width, height = img.size
    if vertical:
        for j in range(1, width // N + 1, 2):
            inverted_part = img.crop((j * N, 0, (j + 1) * N, height))
            inverted_part = ImageOps.invert(inverted_part)
            img.paste(inverted_part, (j * N, 0))
    else:
        for i in range(1, height // N + 1, 2):
            inverted_part = img.crop((0, i * N, width, (i + 1) * N))
            inverted_part = ImageOps.invert(inverted_part)
            img.paste(inverted_part, (0, i * N))
    return img


def mix(img, rules):
    width, height = img.size
    parts = []
    for j in range(3):
        for i in range(3):
            part = img.crop((i * (width // 3), j * (height // 3), (i
+ 1) * (width // 3), (j + 1) * (height // 3)))
            parts.append([part, (i * (width // 3), j * (height //
3))])
```

```
for i in rules:
    img.paste(parts[rules[i]][0], parts[i][1])
return img
```