**отчет**

**по домашнему заданию**

**по дисциплине «Функциональное программирование»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 22Б16 |  | Олизько С.С. |
| Преподаватель |  | Киямов Ж.У. |

**Санкт-Петербург**

**2023 г.**

**Оглавление**

[**1.** **Цель работы** 3](#_Toc144847601)

[**2.** **Задача** 3](#_Toc144847602)

[**3.** **Теоретическая часть** 3](#_Toc144847603)

[**4.** **Алгоритм метода**](#_Toc144847604) 3

[**5.** **Описание программы**](#_Toc144847605) 4

[**6.** **Рекомендации к использованию**](#_Toc144847607) 4

[**7.** **Вывод**](#_Toc144847611) 5

1. **Цель работы:**

Цель данной работы заключается в разработке программы для обработки изображений в многозадачной среде. Программа должна обрабатывать изображения с использованием нескольких фильтров одновременно и обеспечивать безопасное сохранение обработанных изображений в отдельной папке.

1. **Задача:**

Задачей является создание программы, способной обрабатывать изображения параллельно с использованием потоков и нескольких фильтров, таких как сепия, уменьшение размера, размытие, увеличение резкости граней, и сохранять обработанные изображения в выходной папке.

1. **Теоретическая часть:**

Для выполнения данной задачи использованы библиотеки Python, такие как os, threading, tkinter (для создания пользовательского интерфейса), и PIL (Pillow) для обработки изображений. Важно обеспечить безопасный доступ к данным и эффективное управление ресурсами при обработке изображений в многозадачной среде.

1. **Алгоритм метода:**

Шаги, выполняемые программой:

* 1. Определение папки с исходными изображениями и папки для сохранения обработанных изображений.
  2. Создание графического интерфейса с флажками для выбора фильтров (сепия, уменьшение размера, размытие, резкость, увеличение резкости граней).
  3. Пользователь выбирает желаемые фильтры.
  4. Нажатие кнопки "Применить и обработать" инициирует следующие действия: a. Создание списка выбранных фильтров. b. Получение списка имен файлов изображений в папке с исходными изображениями, отфильтрованных по расширениям .jpg, .jpeg и .png. c. Создание потоков для каждого изображения и его обработки. d. В каждом потоке: i. Открытие изображения. ii. Применение выбранных фильтров к изображению. iii. Сохранение обработанного изображения в выходную папку.
  5. Ожидание завершения всех потоков.
  6. Завершение программы.

1. **Описание программы:**
   * process\_image(input\_image, output\_folder, filters): Функция, которая обрабатывает изображение с выбранными фильтрами. Она открывает изображение, применяет фильтры и сохраняет результат в выходную папку.
   * apply\_sepia\_effect(image): Функция, которая применяет эффект сепии к изображению.
   * resize\_image(image): Функция, которая уменьшает размер изображения.
   * apply\_blur(image): Функция, которая применяет размытие к изображению.
   * apply\_sharp(image): Функция, которая увеличивает резкость изображения.
   * apply\_edge(image): Функция, которая увеличивает резкость граней на изображении.
   * main(): Основная функция программы. Она устанавливает пути к папкам с изображениями, создает графический интерфейс, обрабатывает выбор фильтров и запускает обработку изображений в многозадачной среде.
2. **Рекомендации по использованию:**
   * Убедитесь, что установлены библиотеки PIL (Pillow) и tkinter.
   * Разместите исходные изображения в папке "input\_images\_folder" в директории программы.
   * Выберите желаемые фильтры на графическом интерфейсе.
   * Нажмите кнопку "Применить и обработать" для запуска обработки изображений.
3. **Вывод**:

В ходе выполнения работы была разработана программа для параллельной обработки изображений с использованием различных фильтров. Программа обеспечивает безопасное сохранение обработанных изображений и удобный пользовательский интерфейс для выбора фильтров. Она может быть полезной для обработки большого количества изображений, обеспечивая высокую эффективность и удобство в использовании.

GitHub: https://github.com/StephanOlizko/fp\_homework/