Санкт-Петербургский государственный университет

Факультет прикладной математики – процессов управления

**Отчет №4**

по дисциплине «Функциональное программирование»

**Параллельная обработка изображений в многозадачной среде**

Автор работы: Цыбина А.И.

Группа: 22.Б15-пу

Преподаватель: Киямов Ж. У.

Санкт-Петербург, 2023

## Оглавление

[1. Цель работы 3](#_Toc150962359)

[2. Теоретическая часть 3](#_Toc150962360)

[3. Алгоритм метода 3](#_Toc150962361)

[4. Описание программы 4](#_Toc150962362)

[5. Рекомендации программиста 5](#_Toc150962363)

[6. Контрольный пример 5](#_Toc150962364)

[7. Вывод 6](#_Toc150962365)

[8. Список использованной литературы 6](#_Toc150962366)

# Цель работы

Разработать эффективное программное решение для одновременной обработки изображений с применением нескольких фильтров.

# Теоретическая часть

Обработка изображений в многозадачной среде представляет значимую задачу с практической точки зрения. Эффективное воздействие на изображения включает в себя применение различных фильтров, трансформаций и операций, что может быть ресурсоемким процессом. Параллельная обработка изображений способствует оптимизации этого процесса, ускоряя его выполнение.

В данном контексте использование параллельных методов обработки фильтров на изображениях может существенно улучшить скорость обработки по сравнению с последовательным применением каждого фильтра к каждому изображению по очереди. Это приводит к экономии времени и повышению эффективности при обработке большого объема изображений.

Вопрос выбора механизма многозадачности включает в себя использование потоков или процессов. Потоки обеспечивают обработку изображений в пределах одного процесса, что более эффективно с точки зрения затрат памяти. С другой стороны, процессы предоставляют независимую обработку изображений, что актуально при работе с многопроцессорными системами.

Также важно учитывать возможные конфликты доступа к данным при параллельной обработке, особенно в контексте одновременного доступа к файлам изображений для чтения и записи.

# Алгоритм метода

1. Импорт библиотек и определение функций.

Импортируются необходимые библиотеки и их модули: os, threading, PIL, tkinter.

1. Класс приложения-обработчика.

Определяется класс с функциями реализации интерфейса и функционала программы.

1. Создание виджетов.

Определяется функция create\_widgets, которая реализует интерфейс приложения.

1. Фильтры.

Определяется функция get\_filter\_function, которая по названию фильтра описывает соответствующее ему преобразование картинки.

1. Применение фильтров.

Определяется функция apply\_filters, которая применяет выбранные пользователем фильтры к изображению.

1. Параллельная обработка.

Определяется функция process\_images, которая параллельно обрабатывает все изображения из входной директории.

# Описание программы

Программная реализация написана на языке python 3.11 с использованием

следующих пакетов: os, ThreadPoolExecutor, PIL.

Таблица 4.1. Описание функций

|  |  |  |
| --- | --- | --- |
| Имя функции | Аргумент | Описание функции |
| create\_widgets |  | Реализует интерфейс приложения посредством виджетов. |
| browse\_input\_folder |  | Устанавливает выбранную пользователем папку как входную. |
| browse\_output\_folder |  | Устанавливает выбранную пользователем папку как выходную. |
| apply\_filters | Входная папка, выходная папка | Итеративно применяет к изображению выбранные фильтры. |
| get\_filter\_function | Название фильтра | Возвращает преобразование, соответствующее выбранным фильтрам. |
| process\_images |  | Запускает параллельную обработку изображений входной директории. |

# Рекомендации программиста

Для запуска программы необходима 64-битная операционная система Windows, Linux или macOS. Для работы с кодом необходима среда разработки, совместимая с python 3.11 и библиотеки os, threading, PIL.

Ссылка на репозиторий с кодом:

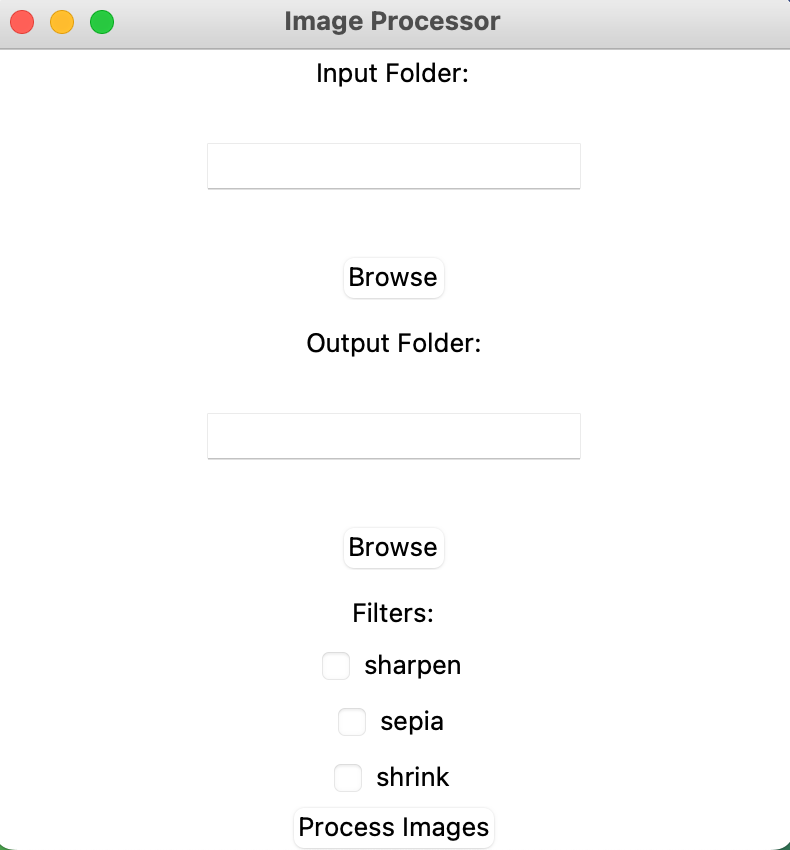
<https://github.com/sastsy/func-prog/tree/main/task4>

# Рекомендации пользователя

В поле Input Folder пользователю следует ввести путь к папке с изображениями для обработки, либо выбрать папку с помощью кнопки “Browse”. Аналогично в поле Output Folder выбирается путь к папке, куда сохранятся изображения с наложенными фильтрами.

Далее пользователь может выбрать от 1 до 3 фильтров: увеличить резкость, применить сепию или сжать изображение.

Кнопка “Process Images” запускает процесс обработки фотографий из Input Folder, после чего они автоматически сохраняются в выбранную директорию Output Folder.



# Контрольный пример

В данном разделе представлен контрольный пример, демонстрирующий

работу программы.



Рисунок 6.1 Фильтр резкости



Рисунок 6.2 Фильтр уменьшения размера



Рисунок 6.3 Фильтра сепии

# Вывод

В ходе выполнения данной работы были изучены основные методы параллельной обработки изображений.

# Список использованной литературы

1. Документация библиотеки os: https://docs.python.org/3/library/os.html
2. Документация библиотеки threading: https://docs.python.org/3/library/threading.html
3. Документация библиотеки PIL:https://pillow.readthedocs.io/en/stable/index.html