Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени

федеральное государственное бюджетное образовательное учреждение высшего образования

МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ

Кафедра «Математической кибернетики и информационных технологий»

Информационные технологии и программирование

Лабораторная работа №9

Выполнила:

студентка группы БВТ2203

Лысикова К. А.

Москва, 2023 г.

**Цель работы:** изучение аннотаций в Java, создание приложения для обработки данных, используя потоки

**Задание**: необходимо разработать приложение, которое считывает данные из исходного источника (например, файл, база данных или сетевой ресурс), применяет к ним различные операции с использованием Stream API, и сохраняет результаты в новый источник данных.

1. Создайте аннотацию @DataProcessor, которая будет использоваться для пометки методов обработки данных.

2. Создайте класс DataManager, который будет отвечать за многопоточную обработку данных. Этот класс должен иметь методы:

* registerDataProcessor(Object processor): Регистрирует объект-
* обработчик данных с аннотацией @DataProcessor.
* loadData(String source): Загружает данные из исходного источника.
* processData(): Запускает многопоточную обработку данных, применяя методы с аннотацией @DataProcessor с использованием Stream API.
* saveData(String destination): Сохраняет обработанные данные в новый источник.

3. Создайте несколько классов, представляющих различные обработчики данных, и пометьте их аннотацией @DataProcessor. Например, можно создать классы для фильтрации, трансформации и агрегации данных. 4. Используйте многопоточность из java.util.concurrent для эффективной обработки данных параллельно.

5. Протестируйте ваше приложение, загрузив данные из исходного источника, применив различные обработчики с помощью Stream API, и сохраните результаты в новый источник.

# **Ход работы**

Создаю аннотацию @DataProcessor, которая будет использоваться для пометки методов обработки данных.

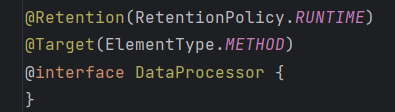


Рисунок 1 - аннотация @DataProcessor

Создаю класс DataManager, который будет отвечать за многопоточную обработку данных. Класс содержит 3 поля: processors (список обработчиков данных), data (список данных), processedData (список обработанных данных). Метод registerProcess добавляет обработчик в список processors. Метод loadData загружает данные из текстового файла.

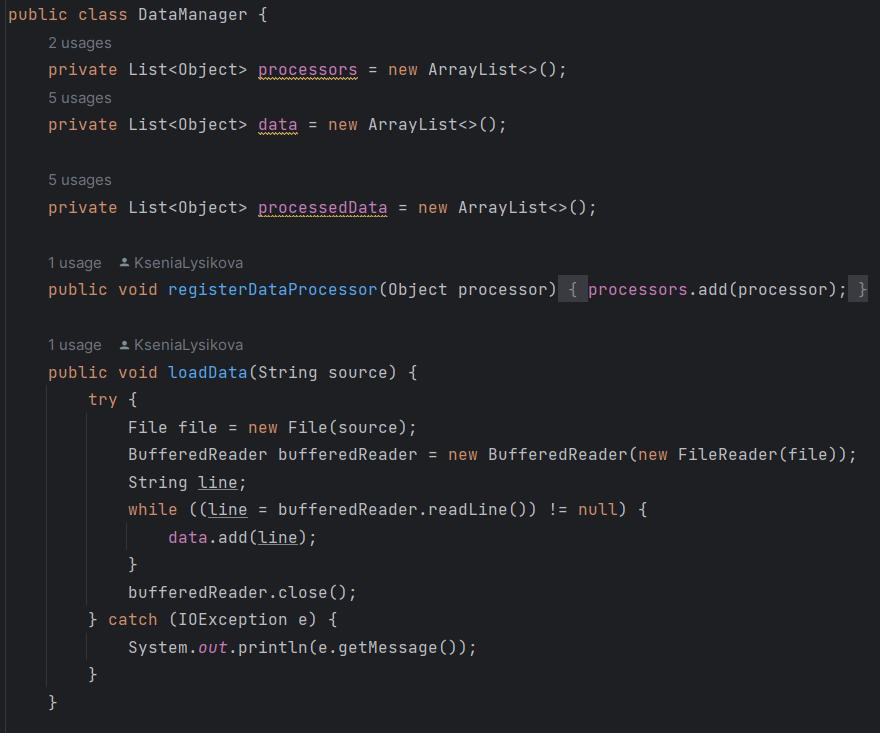


Рисунок 2 - класс DataManager (часть 1)

В методе processData данные обрабатываются с помощью потоков.

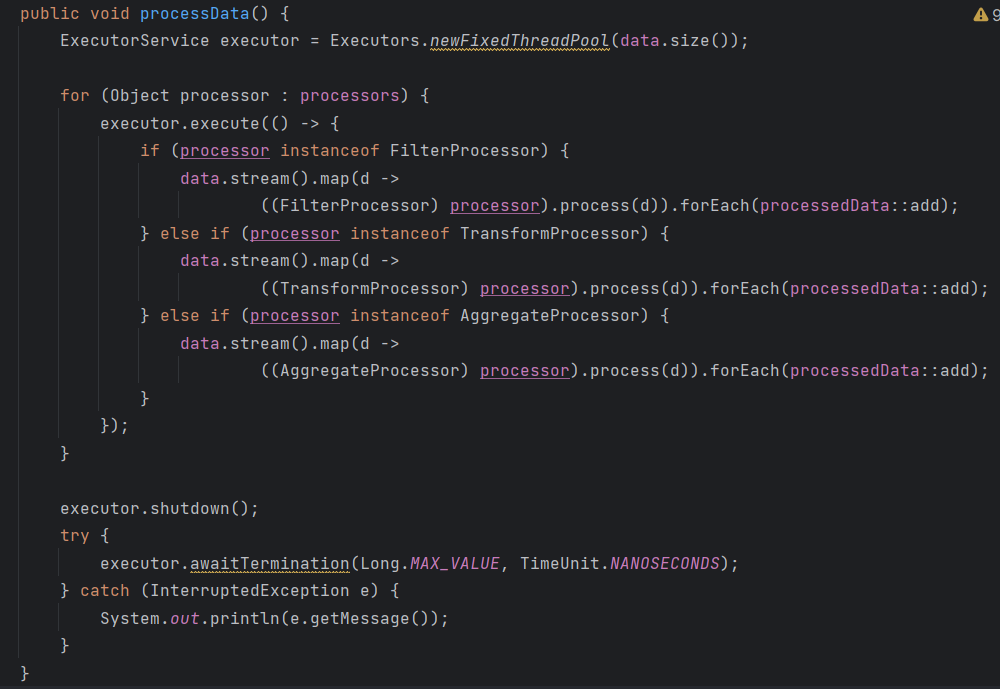


Рисунок 3 - класс DataManager (часть 3)

Метод saveData сохраняет обработанные данные в новый текстовый файл. Класс FilterProcessor отвечает за фильтрацию данных.

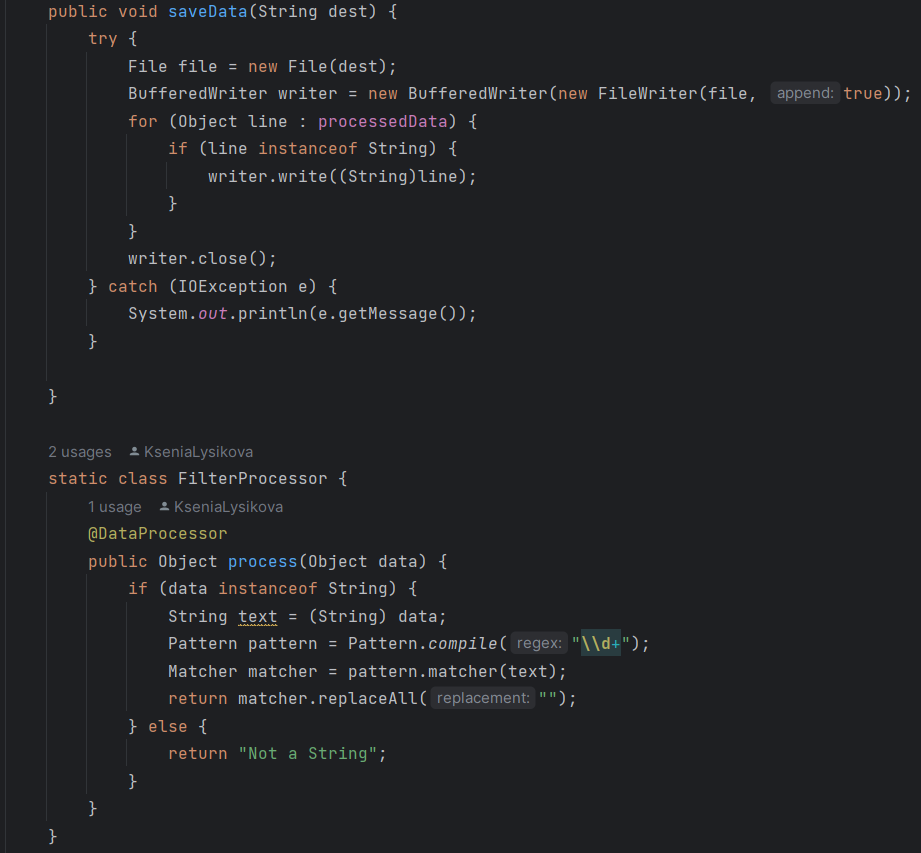


Рисунок 4 – класс DataManager (часть 4)

Класс TransformProcessor отвечает за трансформацию данных, класс AggregateProcessor отвечает за агрегацию данных.



Рисунок 5 - класс DataManager (часть 5)

На рисунке 6 показана работа класса DataManager.

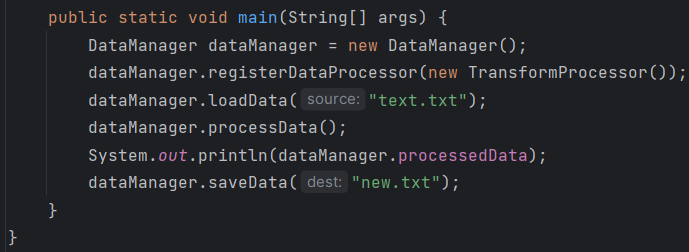


Рисунок 6 - метод main

# **Вывод**

Изучила аннотации в Java. Разработала приложение, которое считывает данные из исходного источника (файл), применяет к ним различные операции с использованием Stream API, и сохраняет результаты в новый источник данных (файл).