ЛР3 Функциональное программирование в Java
Часть 1 – общая

Реализация собственного SplitIterator для обхода строк по символам

В этом задании вам предстоит реализовать собственный Spliterator для итерации по символам строки и использовать его для создания Stream, который позволит обрабатывать строки на уровне отдельных символов.

Требования к заданию

1. **Реализовать класс `CharacterSpliterator`**, который реализует интерфейс `Spliterator<Character>` и позволяет итерацию по символам заданной строки.
    a) **Конструктор** должен принимать строку `String` для обработки.
    b) **Метод `tryAdvance`** должен передавать следующий символ строки в `Consumer<? super Character> action` и продвигаться на один символ вперед.
    c) **Метод `trySplit`** должен реализовывать логику разделения строки на подстроки для параллельной обработки.
    d) **Метод `estimateSize`** должен возвращать количество оставшихся символов.
    e) **Метод `characteristics`** должен возвращать соответствующие характеристики `Spliterator` (например, `ORDERED, SIZED, SUBSIZED, NONNULL, IMMUTABLE`).
2. **Использовать класс `CharacterSpliterator`** для создания `Stream<Character>` с помощью `StreamSupport.stream()`.
3. **Написать программу**, которая:
    a) Читает строку с консоли или использует заранее заданную строку.
    b) Создает `Stream<Character>` из этой строки.
        1. Использует Stream API для выполнения следующих операций: Фильтрация символов: оставить только буквы. Преобразование символов: привести все буквы к верхнему регистру. Подсчет частоты каждого символа (символы в верхнем регистре).

Отчет по первой части

Для отчета по первой части ЛП необходимо приложить скриншот работы программы, чтобы были видны использованные тестовые данные. Приложить листинг только класса итератора (весь код прикладывать не нужно).

Часть 2

В разработанной программе для ЛР1 согласно вариантам заменить все циклы на работу на стримами (поиск, сортировка, сравнение и тд.)

Отчет по второй части

Приложить листинг только одного из репозиториев (классов), где использована работа по стримами. (Но в самой программе изменения необходимо провести везде).

Часть 3

Важно: в списке должно быть минимум 10 объектов с разными значениями параметров.

Варианты

1. Обработка списка книг

Создайте класс Book и список его объектов, затем с помощью Stream API выполните операции над этим списком.

1. **Создать класс Book** с полями:
   - String title — название книги.
   - String author — автор книги.
   - int year — год издания.
   - double price — цена книги.
2. **Создать список объектов** `Book` с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать книги, изданные после 2000 года.
   - Получить список авторов всех книг без повторений.
   - Найти самую дорогую книгу.
   - Подсчитать общее количество книг определенного автора.
   - Отсортировать книги по цене в порядке убывания.
   - Проверить, есть ли в списке книга заданного автора.
   - Сгруппировать книги по авторам.

2. Анализ данных о студентах

Создайте класс Student и список его объектов, затем с помощью Stream API выполните операции над этим списком.

1. **Создать класс Student** с полями:
   - String name — имя студента.
   - int age — возраст студента.
   - double grade — средний балл.
   - List<String> courses — список курсов.
2. **Создать список объектов Student** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать студентов со средним баллом выше 4.5.
   - Получить список всех уникальных курсов, изучаемых студентами.
   - Найти студента с наивысшим средним баллом.
   - Подсчитать количество студентов, изучающих определенный курс.
   - Сгруппировать студентов по возрасту.
   - Проверить, все ли студенты старше 17 лет.
   - Вывести имена студентов, отсортированные в алфавитном порядке.

3. Управление данными о сотрудниках

Создайте класс Employee и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Employee** с полями:
   - String name — имя сотрудника.
   - String department — отдел.
   - double salary — зарплата.
   - int experience — стаж работы (в годах).
2. **Создать список объектов Employee** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать сотрудников с опытом работы более 5 лет.
   - Увеличить зарплату всех сотрудников на 10%.

- Найти сотрудника с минимальной зарплатой.
- Сгруппировать сотрудников по отделам.
- Подсчитать среднюю зарплату по каждому отделу.
- Проверить, есть ли сотрудники с зарплатой выше 100 000.
- Получить список всех отделов без повторений.

4. Работа с продуктами магазина

Создайте класс Product и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Product** с полями:
   - String name — название продукта.
   - String category — категория продукта.
   - double price — цена продукта.
   - int stock — количество на складе.
2. **Создать список объектов Product** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - **a.** Отфильтровать продукты, количество которых на складе меньше 5.
   - **b.** Получить список уникальных категорий продуктов.
   - **c.** Найти самый дешевый продукт в категории "Electronics".
   - **d.** Подсчитать общее количество продуктов на складе.
   - **e.** Отсортировать продукты по названию.
   - **f.** Проверить, есть ли продукты с ценой выше 10 000.
   - **g.** Сгруппировать продукты по категориям и подсчитать количество продуктов в каждой категории.

5. Обработка списка фильмов

Создайте класс Movie и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Movie** с полями:
   - String title — название фильма.
   - String director — режиссер.
   - int releaseYear — год выпуска.
   - double rating — рейтинг фильма.
2. **Создать список объектов Movie** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - **a.** Отфильтровать фильмы с рейтингом выше 8.0.
   - **b.** Получить список режиссеров без повторений.
   - **c.** Найти фильм с наибольшим рейтингом.
   - **d.** Подсчитать количество фильмов, выпущенных после 2010 года.
   - **e.** Отсортировать фильмы по году выпуска в порядке возрастания.
   - **f.** Проверить, есть ли фильмы заданного режиссера.
   - **g.** Сгруппировать фильмы по режиссерам.

6. Анализ данных о транспортных средствах

Создайте класс Vehicle и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Vehicle** с полями:

- String model — модель транспортного средства.
- String type — тип (например, "Car", "Truck", "Motorcycle").
- int year — год выпуска.
- double mileage — пробег.

2. **Создать список объектов Vehicle** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать транспортные средства с пробегом менее 50 000 км.
   - Получить список всех типов транспортных средств без повторений.
   - Найти транспортное средство с наименьшим пробегом.
   - Подсчитать количество транспортных средств старше 5 лет.
   - Отсортировать транспортные средства по году выпуска в порядке убывания.
   - Проверить, все ли транспортные средства имеют пробег меньше 100 000 км.
   - Сгруппировать транспортные средства по типу.

7. Работа с заказами интернет-магазина

Создайте класс Order и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Order** с полями:
   - int orderId — идентификатор заказа.
   - String customer — имя клиента.
   - double amount — сумма заказа.
   - String status — статус заказа ("Pending", "Shipped", "Delivered", "Cancelled").
2. **Создать список объектов Order** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать заказы со статусом "Pending".
   - Найти заказ с наибольшей суммой.
   - Подсчитать общее количество заказов каждого статуса.
   - Сгруппировать заказы по клиентам.
   - Отсортировать заказы по сумме в порядке возрастания.
   - Проверить, есть ли заказы с суммой более 50 000.
   - Получить список уникальных клиентов.

8. Анализ данных о спортсменах

Создайте класс Athlete и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Athlete** с полями:
   - String name — имя спортсмена.
   - String sport — вид спорта.
   - int age — возраст.
   - int medals — количество медалей.
2. **Создать список объектов Athlete** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать спортсменов с количеством медалей больше 5.
   - Получить список всех видов спорта без повторений.
   - Найти самого молодого спортсмена.
   - Подсчитать общее количество медалей по каждому виду спорта.
   - Отсортировать спортсменов по количеству медалей в порядке убывания.
   - Проверить, есть ли спортсмены старше 30 лет.
   - Сгруппировать спортсменов по виду спорта.

9. Обработка данных о клиентах банка

Создайте класс Client и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Client** с полями:
   - String name — имя клиента.
   - String accountType — тип счета ("Savings", "Checking", "Investment").
   - double balance — баланс счета.
   - String city — город проживания.
2. **Создать список объектов Client** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать клиентов с балансом больше 1 000 000.
   - Найти клиента с минимальным балансом.
   - Подсчитать общее количество клиентов в каждом городе.
   - Сгруппировать клиентов по типу счета.
   - Отсортировать клиентов по балансу в порядке возрастания.
   - Проверить, есть ли клиенты с отрицательным балансом.
   - Получить список городов без повторений.

10. Анализ данных о курсах онлайн-обучения

Создайте класс Course и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Course** с полями:
   - String title — название курса.
   - String instructor — инструктор.
   - int duration — длительность в часах.
   - double rating — рейтинг курса.
2. **Создать список объектов Course** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать курсы с рейтингом ниже 4.0.
   - Получить список всех инструкторов без повторений.
   - Найти самый длительный курс.
   - Подсчитать количество курсов каждого инструктора.
   - Отсортировать курсы по рейтингу в порядке убывания.
   - Проверить, есть ли курсы длительностью более 50 часов.
   - Сгруппировать курсы по длительности (например, короткие, средние, длинные).

11. Работа с данными о животных в зоопарке

Создайте класс Animal и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Animal** с полями:
   - String name — имя животного.
   - String species — вид.
   - int age — возраст.
   - double weight — вес.
2. **Создать список объектов Animal** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:

- Отфильтровать животных старше 5 лет.
- Получить список видов животных без повторений.
- Найти самое тяжелое животное.
- Подсчитать количество животных каждого вида.
- Отсортировать животных по весу в порядке возрастания.
- Проверить, все ли животные весят меньше 500 кг.
- Сгруппировать животных по видам.

12. Обработка данных о музыкантах

Создайте класс Musician и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Musician** с полями:
   - String name — имя музыканта.
   - String instrument — инструмент.
   - int albums — количество выпущенных альбомов.
   - String genre — жанр музыки.
2. **Создать список объектов Musician** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать музыкантов с количеством альбомов более 3.
   - Получить список инструментов без повторений.
   - Найти музыканта с наибольшим количеством альбомов.
   - Подсчитать количество музыкантов в каждом жанре.
   - Отсортировать музыкантов по имени в алфавитном порядке.
   - Проверить, есть ли музыканты, играющие на пианино.
   - Сгруппировать музыкантов по инструменту.

13. Анализ данных о произведениях искусства

Создайте класс Artwork и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Artwork** с полями:
   - String title — название произведения.
   - String artist — художник.
   - int year — год создания.
   - String type — тип произведения ("Painting", "Sculpture", "Drawing").
2. **Создать список объектов Artwork** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
   - Отфильтровать произведения, созданные до 1900 года.
   - Получить список художников без повторений.
   - Найти самое старое произведение.
   - Подсчитать количество произведений каждого типа.
   - Отсортировать произведения по году создания в порядке возрастания.
   - Проверить, есть ли произведения заданного художника.
   - Сгруппировать произведения по типу.

14. Работа с данными о рецептах

оздайте класс Recipe и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Recipe** с полями:
    - String name — название рецепта.
    - List<String> ingredients — список ингредиентов.
    - int cookingTime — время приготовления в минутах.
    - String difficulty — сложность ("Easy", "Medium", "Hard").
2. **Создать список объектов Recipe** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
    - Отфильтровать рецепты с временем приготовления менее 30 минут.
    - Получить список всех ингредиентов без повторений.
    - Найти самый сложный рецепт.
    - Подсчитать количество рецептов каждой сложности.
    - Отсортировать рецепты по времени приготовления в порядке возрастания.
    - Проверить, есть ли рецепты, содержащие определенный ингредиент.
    - Сгруппировать рецепты по сложности.

15. Анализ данных о научных публикациях

Создайте класс Publication и список его объектов, затем с помощью Stream API выполните операции над этим списком.

Требования к заданию

1. **Создать класс Publication** с полями:
    - String title — название публикации.
    - String author — автор.
    - int year — год публикации.
    - String journal — название журнала.
2. **Создать список объектов Publication** с различными значениями полей.
3. **Используя Stream API**, выполнить следующие операции:
    - Отфильтровать публикации, выпущенные за последние 5 лет.
    - Получить список журналов без повторений.
    - Найти самую раннюю публикацию.
    - Подсчитать количество публикаций каждого автора.
    - Отсортировать публикации по году в порядке убывания.
    - Проверить, есть ли публикации в заданном журнале.
    - Сгруппировать публикации по журналам.