Практика 30.04

FlatMap - если у нас есть список списков, то можно объединить все списки в один большой. Они будут располагаться в списке последовательно

Редукция.

1. Контейнер для хранения
2. Функция-аккумулятор
3. Функция-комбайнер (конкатенирует два контейнера) (Если у нас работают несколько стримов)

Чем FlatMap отличается от редукции?
FlatMap объединяет стримы. Редукция позволяет работать в произвольными контейнерами.

Collector

Уже написанная редукция для строк.

Что быстрее: цикл или stream?
Цикл быстрее, если стрим однопоточный
Если стримы параллельные, то тут зависит.

К летучке: разобрать и научиться писать цепочки к коллекторами

NB: Никогда не использовать lambda + forEach , если можно написать вместо этого просто for:
Каждый вызов лямбды - новый элемент (на каждый элемент будет создаваться новый объект)

Spliterator

Пам-пам-пам , дописать

Исключения

JVM отслеживает все стеки вызовов. Когда происходит исключение (на процессорном уровне), она должна понять, что это именно жала исключение, зарезать его, сохранить текущий стектрейс, записать исключение и пробросить наверх интеррапт сигнал.

Самое тяжелое - обращаться к jvm , чтобы она собрала стектрейст

Jit не отлавливает исключения, когда оно происходит вываливает в нескомпилированный байт-код, что вредит перфомансу

JAVA Streams

java.util.stream

Стрим — некоторый набор элементов, которые можно обрабатывать скопом.

Особенности:
- Можно не хранить элементы
- Может быть ленивым
- Может быть бесконечным

Когда выпустили Java 8, то там дописали получение стримов почти откуда угодно.
- Из коллекций и массивов
- Из генераторов
- Из файлов
- Случайные потоки
(Любая сущность, которая может быть перечисляемой, получила стрим)

Какие бывают операции над стримами?
- Промежуточные
  - Порождают стрим (т.е. всегда возвращают стрим)
  - Ленивые
- Завершающие
  - Порождают значения
  - Жадные
- Без состояния
  - Не зависят от других элементов
  - map , filter
- С состоянием
  - Зависят от других элементов
- Обрывающие
  - Могут прочесть лишь часть потока
  - limit

Что значит «ленивые»?

Если у нас были какие-то операции над потоком, а потом мы завершили его операцией max(), то из исходного потока мы вытащим ровно столько элементов, сколько нужно, чтобы получить один итоговый.

Стрим итератор - берет начальное значение и функцию, для создания следующий (типа unfoldr)

Стрим генератор - создаёт стрим с помощью лямбды

Конвеерные методы
- filter (JIT несколько фильтров подряд склеет в один)
- skip
- distinct (возвращает стрим без дубликатов)
- map
- peek (как ее нормально использовать????)
- limit
- sortedin
- flatMap (создаёт из каждого элемента потока новый поток и объединяет эти потоки в один)

Терминальные методы
- findFirst
- findAny (когда работаем с параллельными стримами, findFirst использовать неудобно (придётся ждать, пока стрим с первым элементом закончит работу), а тут мы просто берём первый, который попался)
- collect
- count
- anyMatch
- noneMatch
- allMatch
- min
- max
- forEach (терминальный peek)
- forEachOrdered
- toArray()

Пример задачи на летучку, вывести топ-10 слов на

Практика 14.05

Аннотации

- Информация для компилятора
- Доп. Обработка кода во время компиляции
- Могут генерить код во время компиляции

Ограничения:

Дописать

- Метод может возвращать (). Аннотации хранятся полностью в класс-файле, а пользовательский тип в общем случае никак не добавить. Почему тушь можно? Потому что по-факту, это просто строчка + число, а это легко сериализовать.

Использование:
Дописать

Можно делать аргументы по умолчанию

@Target
Указывает, на что будет вешаться аннотация (метод, пакет, т.д.)

@Retention

Метааннотации чтоооо?