1. Какие особенности у `Stream`?

Ленивая инициализация: операции не выполняются до того, как будет вызвана терминальная операция.

Функциональный стиль: предлагает функциональный подход к обработке данных.

Неизменяемость: исходные данные не модифицируются.

Поддержка параллелизма: можно легко конвертировать последовательный поток в параллельный с использованием метода .parallel().

2. Два основных типа операций:

Intermediate (промежуточные): map, filter, sorted и т.д.

Terminal (терминальные): collect, forEach, reduce и т.д.

3. Что будет если дважды вызвать терминальную операцию на `Stream`?

Если дважды вызвать терминальную операцию на одном и том же Stream, будет выброшено исключение IllegalStateException, так как поток становится недоступным после выполнения терминальной операции.

4. Несколько промежуточных операторов:

map

filter

flatMap

distinct

sorted

peek

5. Несколько терминальных операторов:

collect

forEach

reduce

toArray

sum

min

max

6. Как обрабатываются пайплайны(интермдиейт опараторы в `Stream`) сразу на все данные все пайплайны на 1 объект?

Пайплайны обрабатываются лениво. Каждый элемент данных проходит через все промежуточные операторы перед тем, как перейти к следующему элементу, а не через каждый пайплайн полностью сразу.

7. Какая особенность есть у `.sorted(..)`?

Особенность .sorted(): оператор сортировки является stateful, и все элементы должны быть обработаны перед выполнением последующих операций.

8. Все ли операции stateless?

Нет, не все операции являются stateless. Например, sorted и distinct являются stateful операциями.

9. Какое преимущество stateless операций?

Преимущества stateless операций:

Более простые для понимания и реализации.

Легче оптимизировать.

Подходят для параллельной обработки данных.

10. Почему `peek` лучше не использовать для дополнительной логики подсчета?

peek лучше не использовать для дополнительной логики подсчета, потому что это может сделать код менее предсказуемым и усложнить отладку.

11. Дан лист строк "BigBen", "BigBob", "Big", "Ben", "Big Bob". сколько раз объект «Big» встречается в коллекции

Для подсчета количества раз, сколько объект "Big" встречается в коллекции, мы можем использовать следующий код:

java

Copy code

```
List<String> list = Arrays.asList("BigBen", "BigBob", "Big", "Ben", "Big Bob");
long count = list.stream().filter(s -> s.equals("Big")).count();
```

Значение переменной count будет равно количеству вхождений "Big" в коллекцию.