Тестовое задание для стажера на позицию «Разработчик Java»

**Сервис-хранилище типа "ключ-значение"**

Необходимо разработать простой сервис который будет хранить данные в оперативной памяти по текстовому ключу.

https://ru.wikipedia.org/wiki/База\_данных\_«ключ-значение»

**Необходимая функциональность:**

1. Операция чтения (get)

Принимает следующие параметры:

a) ключ для хранилища.

Возвращает данные, хранящиеся по переданному ключу или метку отсутствия данных.

2. Операция записи (set)

Принимает следующие параметры:

a) ключ для хранилища;

b) данные для хранилища, которые будут ассоциированы с переданным ключом;

c) опциональный параметр ttl (продолжительность жизни записи),

по истечении данного временного промежутка данная пара ключ-значение должна автоматически удаляться из хранилища.

Если параметр не передан - использовать ttl по-умолчанию.

Если по переданному ключу уже хранятся данные - их нужно заменять, а также обновлять ttl у данной записи.

Возвращает метку успешности или неуспешности операции.

3. Операция удаления (remove)

Принимает следующие параметры:

a) ключ для хранилища.

Удаляет данные, хранящиеся по переданному ключу.

Возвращает данные, хранившиеся по переданному ключу или метку отсутствия данных.

4. Операция сохранения текущего состояния (dump)

Сохраняет текущее состояние хранилища и возвращает его в виде загружаемого файла.

5. Операция загрузки состояния хранилища (load)

Загружает состояние хранилища из файла, созданного операцией dump (пункт 4).

6 (дополнительно). Библиотека-драйвер, позволяющая работать с вашим удалённым сервисом из java-кода. Что-похожее на драйвера к базам данных.

Т.е. для инициализации библиотека получает адрес+порт вашего сервиса и при вызове её методов обращается к вашему сервису.

**Требования к сервису:**

1. При создании сервиса можно воспользоваться любым из указанных фреймворков: Spring Framework, Spring Boot, или написать приложение на чистой Java.

2. Сервис должен запускаться как standalone-приложение (java -jar service.jar).

3. Необходимо покрыть код unit-тестами. Полнота покрытия остается на усмотрение автора решения.

4. Приложение должно собираться с использованием Maven или Gradle.

5. Общение с сервисом должно осуществляться по протоколу HTTP. По возможности придерживайтесь REST-архитектуры.

6. Не использовать отдельно устанавливаемые базы данных, кэши и т.п.

Можно хранить данные в памяти приложения или использовать простые встраиваемые базы данных (HSQLDB, H2 или Apache Derby).

**Прочие требования:**

1. Язык программирования: Java 8+.

2. Оформление кода должно соответствовать общепринятым нормам (например https://google.github.io/styleguide/javaguide.html).

3. Исходники необходимо упаковать в ZIP-архив вместе с кратким описанием решения и инструкцией по сборке/запуску.

В поставке не должно быть скомпилированных .class-файлов и привязок к среде разработки (.eclipse, .iml и т.п.)

Приветствуется наличие руководства пользователя/документации для api/javadoc для драйвера.

4. Максимальное время на выполнение задания – 2 недели.

5. Всё, что явно не указано в условиях, остаётся на усмотрение автора решения.