Тестовое задание

Проблематика

Представим, что мы разрабатываем систему, позволяющую спланировать для пользователя автобусный маршрут. В данной системе необходимо реализовать специфичную задачу для одной транспортной компании, а именно: фильтровать маршруты между двумя остановками без пересадок. Представим, что у данной транспортной компании очень много длинных маршрутов, и для того, чтобы наша система работала хорошо и быстро, нам необходим сервис, который очень быстро определяет: есть или нет между двумя остановками прямой маршрут без пересадок.

Постановка задачи

Транспортная компания предоставляет файл данных, содержащий список маршрутов. Каждый маршрут, заданный уникальным идентификатором, состоит из списка остановок, которые так же задаются в виде уникальных идентификаторов. Автобусный маршрут определяется последовательностью остановок.
Ваша задача реализовать микро-сервис, который сможет быстро и эффективно ответить: есть ли беспересадочный маршрут, соединяющий две указанные остановки, или нет.
Примечания. Нет необходимости выдавать все возможные маршруты. Нужно только “да” или “нет. Так же, переданные в микросервис идентификаторы остановок, могут в-принципе не быть ни в одном маршруте.

Файл маршрутов

Файл маршрутов представляет из себя текстовый файл. Каждый маршрут задается в *одной* строке. Для каждого маршрута указывается только одна строка.
Каждая строка состоит из последовательности целых положительных чисел. Первое число - идентификатор маршрута, остальные числа - идентификаторы остановок на данном маршруте.
В файле не может содержаться несколько определений одного и того же маршрута.
Остановка может встречаться сразу в нескольких маршрутах, но не может встречаться дважды и более в одном маршруте.
Маршруты однонаправленные, т.е. если из А можно доехать до Б, то это не значит, что этим же маршрутом можно доехать из Б до А.
Идентификаторы маршрутов и остановок имеют допустимый диапазон значений от 1 до 2 147 483 647.
Можно полагать, что в файле будет до 100 000 маршрутов, до 1 000 000 различных остановок и до 1000 остановок в каждом маршруте. Алгоритмы и структуры данных для хранения должны быть выбраны исходя из этих значений. Приложение не должно требовать ОЗУ больше, чем настройки памяти Java по-умолчанию. Разумеется, какая либо СУБД под эти требования не подходит.

REST API

Микросервис должен поддерживать всего один метод, вызываемый по URL: http://localhost:8080/api/direct?from=x&to=y, где “x” - идентификатор начальной остановки, для поиска маршрута, а “y” - конечной. Идентификаторы представлены в виде целых положительных чисел.

Ответом микросервиса должен быть JSON вида:

```
{
      “from”: x,
      “to”: y,
      “direct”: true
}
```

В полях *from* и *to* - значения из запроса, в поле “direct” true, если найден прямой маршрут и false если нет.

Пример данных

Файл с маршрутами:

```
0 0 1 2 3 4
1 3 1 6 5
2 0 6 4
```

Запрос:
http://localhost:8080/api/direct?from=3&to=6

Ответ:

```
{
      “from”: 3,
      “to”: 6,
      “direct”: true
}
```

Требования к реализации

Микросервис должен быть реализован на Java. Можно использовать любые необходимые библиотеки. В этом случае необходимо создать файл для сборки. Оцениваться будет качество кода, скорость его выполнения и достоверность результатов. Наибольшее внимание будет отдаваться реализации хранения данных и поиска, реализация микро-сервиса вторична.