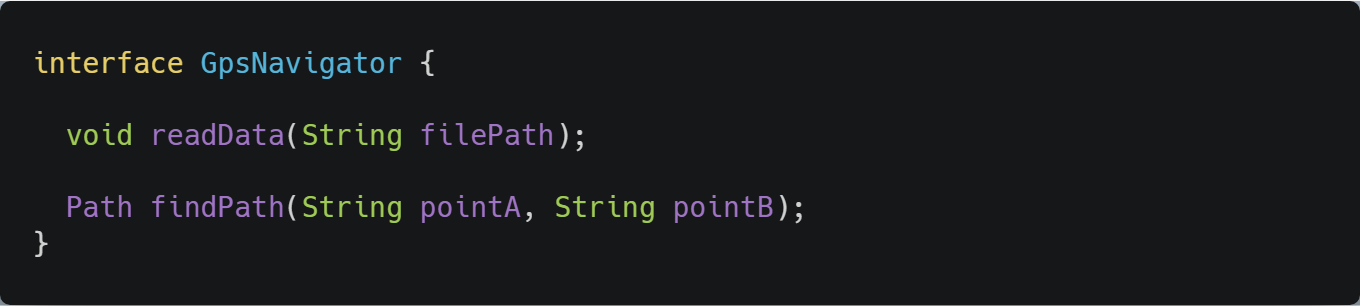
Необходимо разработать простой GPS-навигатор для дальнобойщиков. Результатом работы являться реализация следующего интерфейса:



Где *Path* - POJO удовлетворяющий следующему интерфейсу:



Изменять интерфейсы нельзя. Позволяется создавать дополнительные классы, но только для использования вашей реализацией *GpsNavigator*. **Пользоваться сторонними библиотеками запрещено**.

Описание методов:

*GpsNavigator#readData* принимает на вход путь к файлу, который содержит описание сети дорог страны Б. в следующем формате:

A B 3 2

A C 4 2

C B 5 3

D E 7 4

C D 4 5

D Pizza 7 8

Pizza Metro 9 9

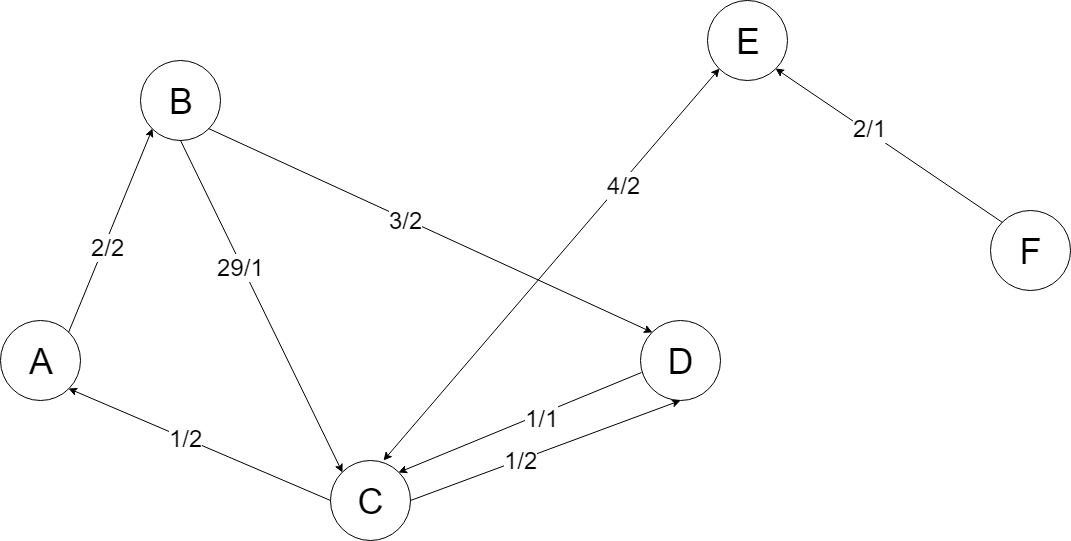
Каждая строка описывает участок дороги с учетом направления. Первые две колонки описывают начальную и конечную точки участка. Имя точки может быть строкой любой длины. Третья колонка – длина участка, четвертая – стоимость проезда по единице длины участка. Колонки разделены одним символом пробела.

*GpsNavigator#findPath* принимает на вход названия двух точек маршрута и ищет самый дешевый путь между ними. Результатом работы данного метода является объект класса Path, который содержит описание пути, а именно:

1. Список точек в порядке их посещения;
2. Суммарную стоимость проезда по пути, которая считается как сумма стоимостей проезда по каждому участку, где стоимость проезда по участку равна произведению длины участка на стоимость проезда по единице длины.

Случаи неверного формата файла, нескольких кратчайших путей, отсутствия пути и прочие должны обрабатываться как исключительные.

Пример сети дорог:



Пример входных данных, описывающих данную сеть:

A B 2 2

C A 1 2

B C 29 1

B D 3 2

C D 1 2

D C 1 1

C E 4 2

E C 4 2

F E 2 1

Примеры запросов и ответов:

findPath(“A”, “C”) -> ABDС, 11

findPath(“C”, “A”) -> CA, 2

findPath(“C”, “F”) -> exception

findPath(“F”, “B”) -> FECAB, 16

В приложении к заданию вы найдёте шаблон проекта, содержащий необходимый интерфейс POJO класс и пример того, как их будем использовать мы.