test_case.md

Дано:

API для работы с базой данных, которая хранит информацию о 20 автомобилях.
https://test.tspb.su/test-task

Поддерживаемые методы:

```
GET /vehicles
POST /vehicles
GET /vehicels/{id}
PUT /vehicles/{id}
DELETE /vehicles/{id}
```

Пример ответа:

```
GET vehicels/1
```

```
{
    "id": 1,
    "name": "Toyota",
    "model": "Camry",
    "year": 2021,
    "color": "red",
    "price": 21000,
    "latitude": 55.753215,
    "longitude": 37.620393
}
```

P.S. Запросы на создание, изменение и удаление ничего не меняют в БД.

Задача:

Написать класс `VehicleManager` для работы с API. Класс должен поддерживать следующий функционал:

- Получение списка автомобилей

- Получение списка автомобилей по заданным параметрам (фильтрация)
- Получение информации об автомобиле по id
- Добавление нового автомобиля
- Изменение информации об автомобиле
- Удаление автомобиля
- Расчет расстояние между двумя автомобилями (в метрах)
- Нахождение ближайшего автомобиля к заданному

Класс `VehicleManager` должен оперировать объектами класса `Vehicle`, который хранит информацию об автомобиле.

Пример использования:

```
>>> from vehicle_manager import VehicleManager, Vehicle
>>> # Создание экземпляра класса VehicleManager
>>> manager = VehicleManger(url="https://test.tspb.su/test-task")
>>> # Получение списка всех автомобилей
>>> manager.get_vehicles()
[<Vehicle: Toyota Camry 2021 red 21000>, ...]
>>> # Получение списка автомобилей, у которых поле name равно 'Toyota'
>>> manager.filter_vehicles(params={"name": "Toyota"})
[<Vehicle: Toyota Camry 2021 red 21000>]
>>> # Получение автомобиля с id=1
>>> manager.get_vehicle(vehicle_id=1)
<Vehicle: Toyota Camry 2021 red 21000>
>>> # Добавление нового автомобиля в базу данных
>>> manager.add_vehicle(
...     vehicle=Vehicle(
...         name='Toyota',
...         model='Camry',
...         year=2021,
...         color='red',
...         price=21000,
            latitude=55.753215,
            longitude=37.620393
...     )
... )
<Vehicle: Toyota Camry 2021 red 21000>
>>> # Изменение информации об автомобиле с id=1
>>> manager.update_vehicle(
...     vehicle=Vehicle(
...         id=1,
...         name='Toyota',
...         model='Camry',
...         year=2021,
...         color='red',
...         price=21000,
            latitude=55.753215,
            longitude=37.620393
...     )
... )
```

```
<Vehicle: Toyota Camry 2021 red 21000>
>>> # Удаление автомобиля с id=1
>>> manager.delete_vehicle(id=1)
>>> # Расчет расстояния между автомобилями с id=1 и id=2
>>> manager.get_distance(id1=1, id2=2)
638005.0864183258
>>> # Нахождение ближайшего автомобиля к автомобилю с id=1
>>> manager.get_nearest_vehicle(id=1)
<Vehicle Kia Sorento 2019 30000>
```

Требования:

- Задание должно быть реализовано на Python 3.6+
- Использование сторонних библиотек, за исключением `requests`, запрещено.
- Решение предоставить в виде ссылки на репозиторий в Github