**Лабораторная работа №2 (Коллекции, понятия класса, интерфейса и объекта класса, разбиение по пакетам, работа с файлами)**

1. Создать класс Персоны, у которой есть поля для ФИО, года рождения, номер телефона, сделать так, чтобы объект этого класса нельзя создавать. Инициализация полей класса должна быть доступна только в конструкторе. Значения полей должны быть доступны через геттеры
2. Создать два наследника класса Персоны – Учитель и Ученик. Учитель должен иметь поля: предмет, часы работы. Ученик должен иметь поля – список изучаемых предметов, хеш-таблица (Map) с средними оценками по предметам. Возможные предметы должны быть оформлены в перечисления (enum) <http://tutorials.jenkov.com/java/enums.html>
3. Научиться переводить объекты Учителя и Ученика со всеми данными внутри в строковое представление (можно в json, см. <https://www.baeldung.com/jackson-object-mapper-tutorial>).
4. Научиться сохранять строковое представление Учителя и ученика в файл. Научиться восстанавливать объект учителя и ученика из файла. На основе этих умений написать класс PeopleDAO (переводится data access object <https://www.baeldung.com/java-dao-pattern>), который умеет в определенной в его конструкторе папке создавать, удалять, обновлять и искать по идентификатору Персоны.
5. Создать CachedPeopleDAO, который умеет то же самое, что и PeopleDAO, но хранит учителей и учеников в памяти в виде Map идентификатор на объект. При этом интерфейс у обоих реализация DAO обязан быть одинаковым.
6. Написать класс PeopleService, который умеет
   1. Создавать объекты учителя и ученика и сохранять из с помощью DAO;
   2. по идентификатору с помощью DAO получать объект и изменять некоторые поля объекта;
   3. по идентификатору удалять объект;
7. Создать класс/объект контроллер, который каждую секунду проверяет наличие файлов «управления» в заданной в его конструкторе папке, если файлы управления есть, то он считывает данные из файла (это команда для PeopleService) и кладет их в очередь https://www.baeldung.com/java-queue. Этот контроллер должен работать в отдельном потоке. Как сделать – читаем здесь: <http://tutorials.jenkov.com/java-concurrency/creating-and-starting-threads.html>. Как только контроллер обработал файлик, файл должен быть удален контроллером.
8. Создать класс/объект диспатчер, который каждую секунду проверяет наличие элементов управления в очереди и, если они есть, то последовательно вытаскивает их и обрабатывает с помощью PeopleService. (еще один отдельный поток).