# Лабораторная работа №3: Еще по структурам данных ОКТМО

Обработка данных средствами Java.

В части 2 желательно несколько раз использовать Stream API Java 8:

* 1. <https://habrahabr.ru/post/213805/>
  2. <https://habrahabr.ru/post/224593/>
  3. <https://habrahabr.ru/company/luxoft/blog/270383/>
  4. <http://info.javarush.ru/translation/2014/06/14/Lambda-выражения-на-примерах.html>

Минимальные требования к использованию Stream API – функции типа map.values().stream().forEach( x-> { … } ) или lst.stream().mapToInt(s->s.length()).max()

Коллекторы и группировка – по желанию.

1. Чтение иерархии муниципальных образований.  
   1. Создайте классы OKTMOGroup для хранения регионов, районов, и т.д. Поля:
      * level типа OKTMOLevel (enum { REGION, RAYON, …}) хранит информацию о уровне деления
      * name
      * code (long)
   2. При чтении файлов .txt или /.csv создавайте объекты OKTMOGroup и добавляйте их в общий Map c ключом long (code) для быстрого поиска или в отдельные Map-ы по уровням (Map в OktmoData, заполняется в функции addGroup(OKTMOGroup) )
      * **Чтение должно использовать регулярные выражения с группами.**
      * Уровень определяется по количеству нулей в конце кода
      * Строки типа «Населенные пункты, входящие в состав….», «Городские поселения…», «Межселенные территории» просто игнорируем, по тексту или коду. Не обязательно включать эту проверку в регулярное выражение, можно обойтись условиями.
      * **Все населенные пункты (без 000 в конце кода) размещаем дополнительно в TreeMap<Long,Place>, где ключом является код.**
      * Признак региона – 000 000 000 в конце кода

"80";"000";"000";"000";"8";"1";"Муниципальные образования Республики Башкортостан";"";"";0;0;14.06.2013;01.01.2014

пусть name="Муниципальные образования Республики Башкортостан", с падежами не разбираемся

* + - Признак района – 000 000 в конце кода, считываем только код и название (Чишминский муниципальный район), ссылку на НП-центр не обязательно.

Пример записи района:  
"80";"657";"000";"000";"4";"1";"Чишминский муниципальный район";"рп Чишмы";"";0;0;14.06.2013;01.01.2014

* + - Города (городские округа) размещаем на том же уровне, разделять с районами не требуется:

"65";"751";"000";"000";"7";"1";"город Нижний Тагил";"г Нижний Тагил";"";0;0;14.06.2013;01.01.2014

* + - Пример записи сельсовета / муниципального образования:

(признак – 000 в конце кода, центры сохранять не требуется)

"80";"657";"405";"000";"2";"1";"Алкинский сельсовет";"с Узытамак";"";0;0;14.06.2013;01.01.2014

* 1. У каждой группы заведите список вложенных групп List<OktmoGroup> , добавляйте каждую встреченную по ходу чтения группу в **последнюю считанную группу уровнем выше** (район в последний регион, например) или **в группу, найденную по коду**.

Хранить список вложенных населенных пунктов можно, но не обязательно.

* 1. Добавьте тесты на количество сельсоветов в районе (или количество районов в регионе) и наличие определённого элемента в списке.

1. Добавьте в класс OktmoAnalyzer методы поиска по существующим структурам данных.
   1. Добавьте метод findAllPlacesInGroup(OKTMOGroup group), возвращающий Stream или List населенных пунктов, которые относятся к данному региону, району или муниципальному образованию.

**Не делайте без необходимости цикл просмотра всего списка НП, их очень много!**

**Способы обойтись без этого:**

* **Ищите подходящие НП по коду в общем TreeMap (может помочь submap)**
* **Делайте вложенные циклы по региону, району и сельсоветам (тогда список ссылок на НП придется хранить внутри сельсоветов).**
* **(усложненный вариант) Объединяйте списки НП по вложенным в регион районам и вложенным в них сельсоветам путем двойного вызова flatMap**

**(он объединяет полученные лямбдой из каждого элемента stream-ы)**

* 1. Добавьте метод поиска самого частого названия населенного пункта findMostPopularPlaceName(String regionName)   
     с быстрым поиском региона по имени (заведите дополнительный Map<String, OKTMOGroup>?) и вызовом предыдущего метода поиска НП.

Желательно сделать вариант с циклом подсчёта количества НП с каждым названием типа

Map<String, AtomicInteger> (у AtomicInteger есть метод увеличения) и сравнить его с Collectors.toMap или Collectors.groupingBy.

* 1. Вывести в другом методе printStatusTableForRegion(String region) таблицу количества НП с каждым статусом в регионе.

(самый короткий вариант – использовать метод из п.1 и метод groupingBy Stream Api, который порождает Map из списков, сгруппированных по заданному вычисляемому ключу)

<https://habrahabr.ru/company/luxoft/blog/270383/>