Ознакомится с нововведениями Java8

<http://docs.oracle.com/javase/8/docs/technotes/guides/collections/changes8.html>

<http://docs.oracle.com/javase/8/docs/technotes/guides/language/enhancements.html#javase8>

<http://www.javacodegeeks.com/2014/05/java-8-features-tutorial.html>

<http://www.javacodegeeks.com/2014/05/java-8-features-tutorial.html#Optional>

[Тагир Валеев. Лекция 7. Элементы функционального программирования на Java](https://www.youtube.com/watch?v=U10maNKqnXg)

[Тагир Валеев. Лекция 8. Stream API](https://www.youtube.com/watch?v=Pk7atYm8bX0)

[Тагир Валеев. Stream API: рекомендации лучших собаководов](https://www.youtube.com/watch?v=vxikpWnnnCU)

[Александр Маторин. Рефакторинг в функциональном стиле](https://www.youtube.com/watch?v=rNg8jrWelXk)

[Сергей Куксенко — Stream API, часть 1](https://www.youtube.com/watch?v=O8oN4KSZEXE)

Реализовать класс похожий на java.util.stream.Stream (<http://www.oracle.com/technetwork/articles/java/ma14-java-se-8-streams-2177646.html>)

Использование этого класса должно выглядеть примерно так:

List<Person> someCollection = ...

Map m = Streams.of(someCollection)

                                      .filter(p -> p.getAge() > 20)

                                      .transform( p -> new Person(p.geAge() + 30)))

                                      .toMap(p -> p.geName(), p -> p);

Streams.of() - статический метод, который принимает коллекцию и создает новый объект Streams

filter() - оставляет в коллекции только те элементы, которые удовлетворяют условию в лямбде.

transform() - преобразует элемент в другой.

toMap - принимает 2 лямбды для создания мапы, в одной указывается, что использовать в качестве ключа, в другой, что в качестве значения.

После выполнения всех операций коллекция someCollection не должна поменяться.

Класс надо параметризовать используя правило PECS

public class Streams<T> {

   public static Streams of(List list) {

            ....

    }

    public Streams filter(........) {

           ...

         return this;

    }

     public Streams transform(........) {

           ...

         return this;

    }

     public Map toMap(........) {

           ...

    }

}