



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА – Российский технологический университет»

**РТУ МИРЭА**

---

---

**Институт информационных технологий (ИТ)  
Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**Дисциплина «Программирование на языке Джава»**

**ОТЧЕТ  
ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №29-30**

Выполнил студент группы ИНБО-02-20

Лукияненко Д.В.

Принял

Степанов П.В.

Практическая работа выполнена «\_\_» \_\_\_\_\_ 2021 г.

«\_\_\_\_\_»  
Отметка о выполнении

«\_\_» \_\_\_\_\_ 2021 г.

**Москва – 2021 г.**

## СОДЕРЖАНИЕ

Цель работы .....	3
Задание .....	3
Репозиторий .....	3
Код выполненной работы .....	4
Тестирование программы .....	5
Вывод .....	6

## Цель работы

Цель данной практической работы – Научися использовать Stream API.

## Задание

Напишите программу, читающую из System.in текст в кодировке UTF-8, подсчитывающую в нем частоту появления слов, и в конце выводящую 10 наиболее часто встречающихся слов.

Словом будем считать любую непрерывную последовательность символов, состоящую только из букв и цифр. Например, в строке "Мама мыла раму 33 раза!" ровно пять слов: "Мама", "мыла", "раму", "33" и "раза".

Подсчет слов должен выполняться без учета регистра, т.е. "МАМА", "мама" и "Мама" — это одно и то же слово. Выводите слова в нижнем регистре.

Если в тексте меньше 10 уникальных слов, то выводите сколько есть.

Если в тексте некоторые слова имеют одинаковую частоту, т.е. их нельзя однозначно упорядочить только по частоте, то дополнительно упорядочите слова с одинаковой частотой в лексикографическом порядке.

Задача имеет красивое решение через стримы без циклов и условных операторов. Попробуйте придумать его.

## Репозиторий

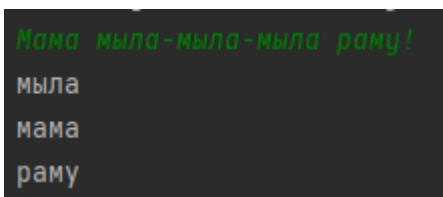
Ссылка: <https://github.com/neluckoff/mirea-java-lessons/tree/master/src/ru/luckoff/mirea/exercies29and30>

## Код выполненной работы

```
public class Main {  
    public static void main(String[] args) throws IOException {  
        Comparator<Map.Entry<String, Integer>> c1 = Map.Entry.comparingByValue(Comparator.reverseOrder());  
        Comparator<Map.Entry<String, Integer>> c2 = Map.Entry.comparingByKey();  
  
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
  
        String str = in.readLine().toLowerCase().replaceAll("([!,.])", "");  
        String[] words = str.split("\\s-");  
  
        Arrays.stream(words).stream<String>  
            .map(String::toLowerCase)  
            .collect(Collectors.groupingBy(x -> x, Collectors.summingInt(p -> 1))) Map<String, Integer>  
            .entrySet() Set<Map<K, V>.Entry<String, Integer>>  
            .stream() Stream<Map<K, V>.Entry<String, Integer>>  
            .sorted(c1.thenComparing(c2))  
            .map(Map.Entry::getKey) Stream<String>  
            .limit(10)  
            .forEach(System.out::println);  
    }  
}
```

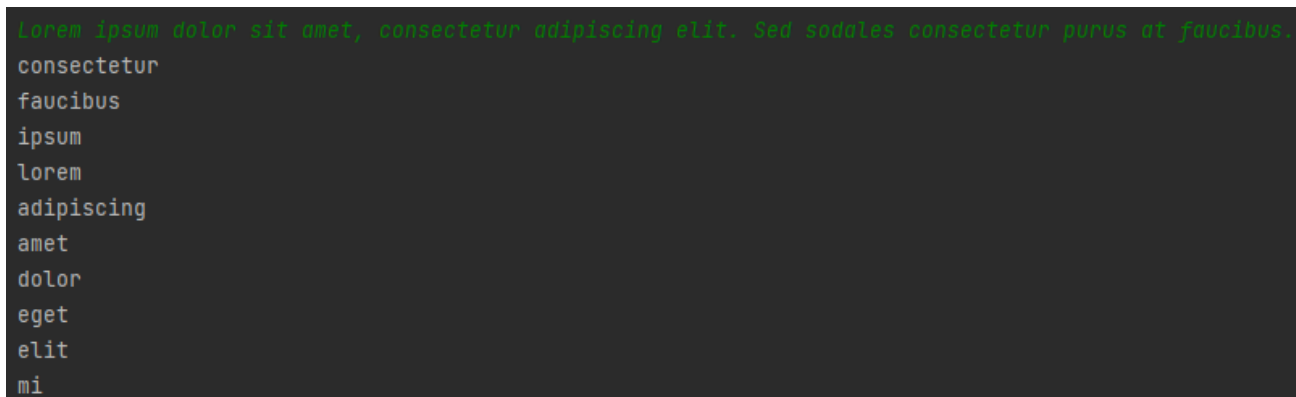
Рисунок 1 – Реализация программы

# Тестирование программы



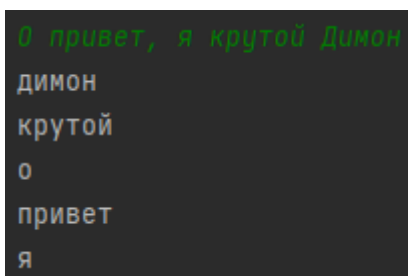
```
Мама мыла-мыла-мыла раму!  
мыла  
мама  
раму
```

Рисунок 2 – Первое тестирование



```
lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sodales consectetur purus at faucibus.  
consectetur  
faucibus  
ipsum  
lorem  
adipiscing  
amet  
dolor  
eget  
elit  
mi
```

Рисунок 3 – Второе тестирование



```
О привет, я крутой Димон  
димон  
крутой  
о  
привет  
я
```

Рисунок 4 – Третье тестирование

## **Вывод**

В результате выполнения данной практической работы мною был изучен Stream API, благодаря которому я смог выполнить поставленную задачу без использования циклов и условных операторов.