|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01/07 **Интеллектуальные системы анализа, обработки и интерпретации больших данных**

**Отчет**

|  |
| --- |
| **по лабораторной работе № 7** |

**Название:**

Строки и регулярные выражения

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-23М |  |  | Т.А. Малкина |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2022

Вариант:

1. В каждом слове текста k-ю букву заменить заданным символом. Если k больше длины слова, корректировку не выполнять.
2. В русском тексте каждую букву заменить ее порядковым номером в алфавите. При выводе в одной строке печатать текст с двумя пробелами между буквами, в следующей строке внизу под каждой буквой печатать ее номер.
3. В тексте найти все пары слов, из которых одно является обращением другого.
4. Найти и напечатать, сколько раз повторяется в тексте каждое слово, которое встречается в нем.
5. В тексте нет слов, начинающихся одинаковыми буквами. Напечатать слова текста в таком порядке, чтобы последняя буква каждого слова совпадала с первой буквой последующего слова. Если все слова нельзя напечатать в таком порядке, найти такую цепочку, состоящую из наибольшего количества слов.
6. Найти наибольшее количество предложений текста, в которых есть одинаковые слова.
7. Все слова английского текста рассортировать по возрастанию количества заданной буквы в слове. Слова с одинаковым количеством расположить в алфавитном порядке.
8. Ввести текст и список слов. Для каждого слова из заданного списка найти, сколько раз оно встречается в тексте, и рассортировать слова по убыванию количества вхождений.

Решение:

|  |
| --- |
| package com.company;  import java.util.\*;  public class Main {  public static void main(String[] args) {  /\*  Вариант 1  1. В каждом слове текста k-ю букву заменить заданным символом.  Если k больше длины слова, корректировку не выполнять.  2. В русском тексте каждую букву заменить ее порядковым номером в алфавите.  При выводе в одной строке печатать текст с двумя пробелами между буквами, в следующей строке внизу под каждой буквой печатать ее номер.   \*/  System.out.println("------------------------------");  System.out.println("№ 1.1");  int k = 2;  char symbol = '@';  String text1 = "Truth will out. Still waters run deep.";  System.out.println(text1.replaceAll(String.format("(\\b\\p{L}{%d})\\p{L}", k - 1), "$1" + symbol));  System.out.println("------------------------------");  System.out.println("№ 1.2");  String text2 = "При выводе в одной строке печатать текст с двумя пробелами между буквами, в следующей строке внизу под каждой буквой печатать ее номер.";  for (char letter : text2.toCharArray()) {  System.out.print(" "+letter+" ");  }  System.out.println();  List<String> rus\_letters = new ArrayList(Arrays.asList('а', 'б', 'в', 'г', 'д', 'е', 'ё', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я'));  String text22 = text2.toLowerCase();  for (char letter : text22.toCharArray()) {  if (letter !=' '){  System.out.print(" "+(rus\_letters.indexOf(letter)+1)+" ");  }  else {  System.out.print(" ");  }  }  System.out.println();  /\*  Вариант 2  1. В тексте найти все пары слов, из которых одно является обращением другого.  2. Найти и напечатать, сколько раз повторяется в тексте каждое слово, которое встречается в нем.  \*/  System.out.println("------------------------------");  System.out.println("№ 2.1");  String text\_21 = "слово палиндром собака птица оволс лвысо акабос";  String[] words = text\_21.split(" ");  String obr ="";  for(String word:words) {  StringBuffer buffer = new StringBuffer(word);  buffer.reverse();  if (text\_21.contains(buffer) & !(obr.contains(buffer))){  obr += word;  obr += buffer;  System.out.println("Пара :"+word+" - "+buffer);  }  }  System.out.println("------------------------------");  System.out.println("№ 2.2");  String text\_22 = "слово раз два три слово три три пара";  String[] words2 = text\_22.split(" ");  Map<String, Integer> counterMap = new HashMap<>();  for (String word : words2) {  if(!word.isEmpty()) {  Integer count = counterMap.get(word);  if(count == null) {  count = 0;  }  counterMap.put(word, ++count);  }  }  for(String word : counterMap.keySet()) {  System.out.println(word + ": " + counterMap.get(word));  }  /\*  Вариант 3  1. В тексте нет слов, начинающихся одинаковыми буквами.  Напечатать слова текста в таком порядке, чтобы последняя буква каждого слова совпадала с первой буквой последующего слова.  Если все слова нельзя напечатать в таком порядке, найти такую цепочку, состоящую из наибольшего количества слов.   2. Найти наибольшее количество предложений текста, в которых есть одинаковые слова.  \*/  System.out.println("------------------------------");  System.out.println("№ 3.1");  String text\_31 = "aбв моргн горк вапсм рог нпр";  String[] words31 = text\_31.split(" ");  Map<String, Integer> counterMap31 = new HashMap<>();  int k1 =0;  for (String word : words31) {  if(!word.isEmpty()) {  counterMap31.put(word.substring(0,1),k1);  k1++;  System.out.println(word+":: "+word.substring(0,1)+","+word.substring(word.length()-1));  }  }  String[] word\_change = new String[words31.length] ;  int j=0;  for (int i = 0; i < words31.length-1; i++){  if(!words31[i].isEmpty()) {  String a = words31[i].substring(words31[i].length()-1);  if (counterMap31.containsKey(a)){  word\_change[j] = words31[i];  word\_change[j+1] = words31[counterMap31.get(a)];  words31[counterMap31.get(a)] = " ";  j+=2;  }  }  }  for (int i = 0; i < word\_change.length; i++){  System.out.print(word\_change[i]+" ");  }  System.out.println();  System.out.println("------------------------------");  System.out.println("№ 3.2");  String text\_32 = "слово раз два три слово три три пара. пропнго. слово раз два три слово три три пара. слово раз два три слово три три пара. 123434выкпачю. слово слово. рв ке рг!";  String[] strings32 = text\_32.split("[\\.\\!\\?]");  int count\_str=0;  boolean flag = false;  for(String str : strings32) {  String[] words32 = str.split(" ");  Map<String, Integer> counterMap32 = new HashMap<>();  for (String word : words32) {  if(!word.isEmpty()) {  Integer count = counterMap32.get(word);  if(count == null) {  count = 0;  }  counterMap32.put(word, ++count);  }  }  for (Integer value : counterMap32.values()) {  if (value > 1){  flag = true;  }  }  if (flag){  count\_str +=1;  flag = false;  }  }  System.out.println("количество предложений текста, в которых есть одинаковые слова - "+count\_str);  /\*  Вариант 4  1. Все слова английского текста рассортировать по возрастанию количества заданной буквы в слове.  Слова с одинаковым количеством расположить в алфавитном порядке.  2. Ввести текст и список слов. Для каждого слова из заданного списка найти, сколько раз оно встречается в тексте,  и рассортировать слова по убыванию количества вхождений.  \*/  System.out.println("------------------------------");  System.out.println("№ 4.1");  String text\_41 = "aaaaaaaagtraaa artg attttta aaaaq aaaafghhhn";  String[] words41 = text\_41.split(" ");  Map<String, Integer> counterMap41 = new HashMap<>();  for (String word : words41) {  if(!word.isEmpty()) {  char[] tmp = word.toCharArray();  int count = 0;  for (char ch : tmp){  if (ch=='a'){  count++;  }  }  counterMap41.put(word,count);  }  }  String[] word\_change4 = new String[words41.length];  for (String word : words41) {   }   for (int i = 0; i < words41.length; i++){  if (counterMap41){  }  }   for (String word : words41) {  System.out.println(word+" - "+ counterMap41.get(word));  }    } } |

Ссылка на репозиторий:

https://github.com/Carpediem2026/BigData