|  |  |
| --- | --- |
| **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 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений**

**Отчет**

**по лабораторной работе № 7**

**Название:** Строки. Регулярные выражения

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

Студент гр. ИУ6-23М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_** Г. Е. Горский **\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**П.В. Степанов **\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

**Цель:** ознакомиться с базовыми принципами языка Java для работы с большими данными

**Вариант 1:**

1. В зависимости от признака (0 или 1) в каждой строке текста удалить указанный символ везде, где он встречается, или вставить его после k-гo символа.
2. Из небольшого текста удалить все символы, кроме пробелов, не являющиеся буквами. Между последовательностями подряд идущих букв оставить хотя бы один пробел.

**Решение:**

public class Main {  
  
 public static void main(String[] args) {  
 char ch ='a';  
 String str = "sdfsf dsfa asd adasd asd";  
 int k=0; //признак  
 int k1=5; //позиция K  
 if(k==0) {  
 System.out.println("sdfsf dsfa asd adasd asd".replaceAll("[-" + ch + "]", ""));  
 }  
 k=1;  
 if (k==1) {  
 StringBuilder stringBuilder = new StringBuilder();  
 stringBuilder.append(str).insert(k1,ch);  
 System.out.println(stringBuilder);  
 }  
 }  
}

Результат выполнения представлен на рисунке 1.

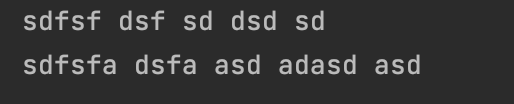
****

Рисунок 1 — пример выполнения

public class Main {  
 public static void main(String[] args) {  
 System.out.println("Hello world!".replaceAll("[^\\w\\s]", ""));  
 }  
}

Результат выполнения представлен на рисунке 2.

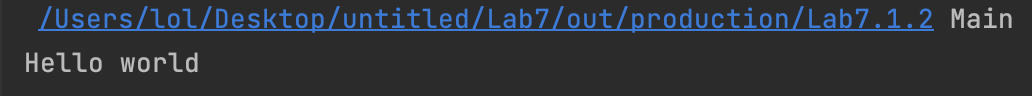


Рисунок 2 — пример выполнения

**Вариант 2:**

1. В тексте найти и напечатать все слова максимальной и все слова минимальной длины.
2. Напечатать квитанцию об оплате телеграммы, если стоимость одного слова задана.

**Решение:**

import java.util.Comparator;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class Main {  
  
 public static void main(String[] args) {  
 System.out.println("Hello hello world!");  
 List<String> strings = List.of("Hello helloasdasd world! wer wed".split("\\s"));  
 System.out.println("min:");  
 strings.forEach((o1) -> {  
 if(o1.length()==strings.stream().min(Comparator.comparing(String::length)).get().length()){  
 System.out.print(o1+" ");  
 }  
 });  
 System.out.println("\nmax:");  
 strings.forEach((o1) -> {  
 if(o1.length()==strings.stream().max(Comparator.comparing(String::length)).get().length()){  
 System.out.print(o1+" ");  
 }  
 });  
 }  
}

Результат выполнения представлен на рисунке 3.

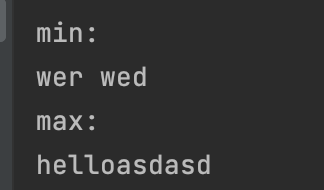


Рисунок 3 — пример выполнения

public class Main {  
 public static void main(String[] args) {  
 int k=2;  
 String str="Token holders are an important part of web3 protocols as they play a crucial role in the functioning and success of the network.";  
 System.out.println("стоимость одного слова:" + k+"\nВаш текст: "+ str);  
 System.out.println("\nСтоимость равна:" + str.split("\\s").length\*k);  
 }  
}

Результат выполнения представлен на рисунке 4.

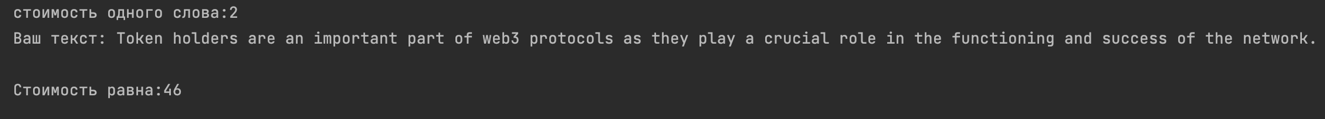


Рисунок 4 — пример выполнения

**Вариант 3:**

1. Текст шифруется по следующему правилу: из исходного текста выбирается 1, 4, 7, 10-й и т.д. (до конца текста) символы, затем 2, 5, 8, 11-й и т.д. (до конца текста) символы, затем 3, 6, 9, 12-й и т.д. Зашифровать заданный текст.
2. На основании правила кодирования, описанного в предыдущем примере, расшифровать заданный набор символов.

**Решение:**

import java.util.Comparator;  
  
public class Main {  
 public static void main(String[] args) {  
 System.out.println("Hello world!");  
 StringBuilder stringBuilder1= new StringBuilder();  
 StringBuilder stringBuilder2= new StringBuilder();  
 StringBuilder stringBuilder3= new StringBuilder();  
 String str = "Hello world!";  
 for (int i=3;i<str.length()+3; i+=3){  
 stringBuilder1.append(str.charAt(i-3));  
 stringBuilder2.append(str.charAt(i-2));  
 stringBuilder3.append(str.charAt(i-1));  
 }  
 System.out.println("1: "+ stringBuilder1);  
 System.out.println("2: "+ stringBuilder2);  
 System.out.println("3: "+ stringBuilder3);  
 }  
}

Результат выполнения представлен на рисунке 5.

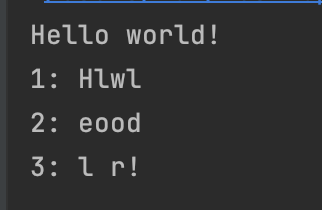


Рисунок 5 — пример выполнения

public class Main {  
 public static void main(String[] args) {  
 System.out.println("Hello world!");  
 StringBuilder stringBuilder1= new StringBuilder();  
 StringBuilder stringBuilder2= new StringBuilder();  
 StringBuilder stringBuilder3= new StringBuilder();  
 StringBuilder stringBuilder4 = new StringBuilder();  
 String str = "Hello world!";  
 for (int i=3;i<str.length()+3; i+=3){  
 stringBuilder1.append(str.charAt(i-3));  
 stringBuilder2.append(str.charAt(i-2));  
 stringBuilder3.append(str.charAt(i-1));  
 }  
 System.out.println(str.length()+3);  
 System.out.println("1: "+ stringBuilder1);  
 System.out.println("2: "+ stringBuilder2);  
 System.out.println("3: "+ stringBuilder3);  
 //расшифровка  
 for (int i=0; i<(stringBuilder1.length()+stringBuilder2.length()+stringBuilder3.length())/3; i++){  
 System.out.print(""+stringBuilder1.charAt(i)+ stringBuilder2.charAt(i) + stringBuilder3.charAt(i));  
 }  
 }  
}

Результат выполнения представлен на рисунке 6.

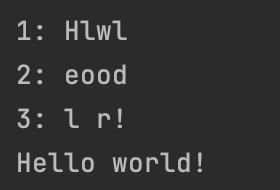


Рисунок 6 — пример выполнения

**Вариант 4:**

1. Подсчитать, сколько слов в заданном тексте начинается с прописной буквы.
2. Подсчитать, сколько раз заданное слово входит в текст.

**Решение:**

import java.util.ArrayList;  
import java.util.HashMap;  
import java.util.List;  
import java.util.concurrent.atomic.AtomicInteger;  
  
public class Main {  
 public static void main(String[] args) {  
 AtomicInteger j= new AtomicInteger();  
 List<String> strings = List.of("Hello helloasdasd world! gWer wed gWer".split("\\s"));  
 strings.forEach((o1) -> {  
 if(o1.matches("[^A-Z](.+)")){  
 System.out.print(o1+" ");  
 j.getAndIncrement();  
 }  
 });  
  
 System.out.println("\nCлов в заданном тексте с прописной буквы: " + j);  
 System.out.println("----------------------------------");  
 //8. Подсчитать, сколько раз заданное слово входит в текст.  
 String word1="gWer!"; //заданное слово  
 int count=0; //количество вхождений  
 for (String word : strings) {  
 System.out.println(word);  
 if (word1.contains(word)) {  
 count++;  
 }  
 }  
 System.out.println(word1+" : "+ count);  
 }  
}

Результат выполнения представлен на рисунке 7.

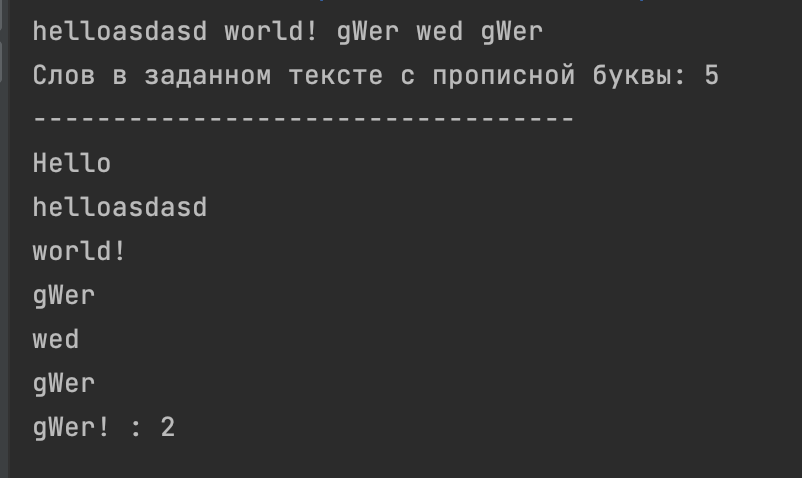


Рисунок 7 — пример выполнения

**Вывод:** были разработаны классы и методы согласно вариантам.