|  |  |
| --- | --- |
| **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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

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

**Название:**

Строки. Регулярные выражения.

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | Е.Е. Шморгун |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Вариант 1 Задача 9.** Из текста удалить все слова заданной длины, начинающиеся на согласную букву.

Main.java

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Main {

public static void main(String[] args) {

int n = 4;

String regex = String.format("(\\b[b-df-hj-np-tv-z][a-z]{%s})\\b", n - 1);

Pattern pattern = Pattern.compile(regex, Pattern.CASE\_INSENSITIVE);

String text = "There is VERY nice arch in center.";

Matcher matcher = pattern.matcher(text);

System.out.println(matcher.replaceAll(""));

}

}

**Вариант 1 Задача 10.** Удалить из текста его часть, заключенную между двумя символами, которые вводятся (например, между скобками ‘(’ и ‘)’ или между звездочками ‘\*’ и т.п.).

Main.java

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

String special\_chars\_regex = "([\*\\.|()\\[\\]])";

System.out.println("Enter start symbol");

String start = in.nextLine();

System.out.println("Enter end symbol");

String end = in.nextLine();

String start\_reg = Pattern

.compile(special\_chars\_regex)

.matcher(start)

.replaceFirst("\\\\$1");

String end\_reg = Pattern

.compile(special\_chars\_regex)

.matcher(end)

.replaceFirst("\\\\$1");

String regex = String.format("%s(.+)%s", start\_reg, end\_reg);

System.out.println(regex);

String text = String.format("There is %ssome%s te\*xt\*", start, end);

System.out.println("Source text:");

System.out.println(text);

System.out.println("Result text:");

System.out.println(deleteGroup(regex, text));

}

public static String deleteGroup(String regex, String source) {

Matcher m = Pattern.compile(regex).matcher(source);

m.find();

return new StringBuilder(source)

.replace(m.start(1), m.end(1),"")

.toString();

}

}

**Вариант 2 Задача 9.** В стихотворении найти одинаковые буквы, которые встречаются во всех словах.

Main.java

import java.util.Arrays;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import java.util.concurrent.atomic.AtomicReference;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) {

String word\_regex = "\\b";

String text = "Nature's first green is gold,\n" +

"Her hardest hue to hold.\n" +

"Her early leaf's a flower;\n" +

"But only so an hour.\n";

List<String> words = List.of(text.split(word\_regex));

Set<Character> characters = new HashSet<>();

char[] text\_chars = text.toCharArray();

for (int i = 0; i < text\_chars.length; i++) {

Character character = new Character(text\_chars[i]);

System.out.print(character);

characters.add(character);

}

Set<Character> presented = characters.stream().filter(character -> {

AtomicReference<Boolean> char\_present = new AtomicReference<>(true);

words.forEach(word -> {

Set<Character> word\_characters = new HashSet<>();

char[] word\_chars = word.toCharArray();

for (int i = 0; i < word\_chars.length; i++) {

Character word\_character = new Character(word\_chars[i]);

word\_characters.add(word\_character);

}

if (!word\_characters.contains(character)) {

char\_present.set(false);

}

});

return char\_present.get();

}).collect(Collectors.toSet());

System.out.println("Results:");

System.out.println(presented);

}

}

**Вариант 2 Задача 10.** В тексте найти первую подстроку максимальной длины, не содержащую букв.

Main.java

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Main {

public static void main(String[] args) {

String regex = "([^a-zA-Z]+)";

String text = "Current population is 134 millions, but it will be about 1000 millions";

Matcher matcher = Pattern.compile(regex).matcher(text);

CharSequence max\_seq = null;

while (matcher.find()) {

CharSequence sequence = text.subSequence(matcher.start(), matcher.end());

if (max\_seq == null || max\_seq.length() < sequence.length()) {

max\_seq = sequence;

}

}

System.out.print(max\_seq);

}

}

**Вариант 3 Задача 9.** Напечатать слова русского текста в алфавитном порядке по первой букве. Слова, начинающиеся с новой буквы, печатать с красной строки.

Main.java

import java.util.\*;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) {

String text = "Это очень важный и ответсвенно написаный тестовый текст, который надо будет обработать.";

String cleaned\_text = text.replaceAll("[^а-яА-Я ]", "");

List<String> words = Arrays.asList(cleaned\_text.split(" "));

words.sort(Comparator.comparingInt(a -> a.toLowerCase().charAt(0)));

System.out.println(words);

for (int i = 0; i < words.size() - 1; i++) {

System.out.print(words.get(i));

if (words.get(i + 1).charAt(0) != words.get(i).charAt(0)) {

System.out.print("\n");

} else {

System.out.print(" ");

}

}

System.out.print(words.get(words.size() - 1));

}

}

**Вариант 3 Задача 10.** Рассортировать слова русского текста по возрастанию доли гласных букв (отношение количества гласных к общему количеству букв в слове)

Main.java

import java.util.Arrays;

import java.util.List;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Main {

public static void main(String[] args) {

String text = "Дамы и господа, пример текста на русском языке";

String cleaned\_text = text.replaceAll("[^а-яА-Я ]", "");

List<String> words = Arrays.asList(cleaned\_text.split(" "));

words.sort((a, b) -> {

if (calc\_word\_ratio(a) > calc\_word\_ratio(b)) {

return 1;

} else {

return -1;

}

});

System.out.println(words);

}

public static double calc\_word\_ratio(String word) {

String vowel\_regex = "[аиоеяуыюАИОЕЯУЫЮ]";

Matcher vowel\_matcher = Pattern.compile(vowel\_regex).matcher(word);

double vowel\_count = 0;

while (vowel\_matcher.find()) vowel\_count += 1;

return (vowel\_count / word.length());

}

}

**Вариант 4. Задача 9.** Преобразовать каждое слово в тексте, удалив из него все последующие (предыдущие) вхождения первой (последней) буквы этого слова

Main.java

import java.util.Arrays;

import java.util.List;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import java.util.stream.Collectors;

public class Main {

public static void main(String[] args) {

String text = "Дамы и господа, пример текста на русском языке. Слово, Ананас";

String cleaned\_text = text.replaceAll("[^а-яА-Я ]", "");

List<String> words = Arrays.asList(cleaned\_text.split(" "));

List<String> parsed\_words = words.stream().map(word -> {

if (word.length() < 3) {

return word;

} else {

char begin = word.charAt(0);

char end = word.charAt(word.length() - 1);

String regex = String.format(

"[%c%c%c%c]",

Character.toLowerCase(begin),

Character.toUpperCase(begin),

Character.toLowerCase(end),

Character.toUpperCase(end)

);

String parsed = begin + word.replaceAll(regex, "") + end;

return parsed;

}

}).collect(Collectors.toList());

System.out.println(parsed\_words);

}

}

**Вариант 4. Задача 10.** Исключить из текста подстроку максимальной длины, начинающуюся и заканчивающуюся одним и тем же символом

Main.java

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Main {

public static void main(String[] args) {

String text = "Some text for testing";

char[] text\_chars = text.toCharArray();

String best\_regex = null;

Integer max\_len = null;

for (int i = 0; i < text\_chars.length; i++) {

char seq\_begin = text\_chars[i];

String regex = String.format("%c.+%c", seq\_begin, seq\_begin);

Matcher matcher = Pattern.compile(regex).matcher(text);

while (matcher.find()) {

int len = matcher.end() - matcher.start();

if (max\_len == null || max\_len < len) {

max\_len = len;

best\_regex = regex;

}

}

}

System.out.println(text.replaceAll(best\_regex, ""));

}

}

**Вывод:** В данной работе я изучил работу со строками, основные методы и принципы. А также узнал о основных классах и методах для работы с регулярными выражениями.