|  |  |
| --- | --- |
| 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**

**Вариант № 3**

**Название:** Строки и регулярные выражения

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

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

Москва, 2023

**Цель:** изучить и освоить принципы работы со строками и регулярными выражениями в Java.

**Задание 1:** В тексте после буквы Р, если она не последняя в слове, ошибочно напечатана буква А вместо О. Внести исправления в текст.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 String sourceFilePath = "input.txt";  
 String outputFilePath = "output.txt";  
  
 // Шаблон для поиска ошибок с учетом регистра и условия не быть в конце слова  
 Pattern pattern = Pattern.compile("РА(?=\\w)", Pattern.CASE\_INSENSITIVE | Pattern.UNICODE\_CHARACTER\_CLASS);  
  
 try (Scanner scanner = new Scanner(new File(sourceFilePath));  
 PrintWriter writer = new PrintWriter(outputFilePath)) {  
 while (scanner.hasNextLine()) {  
 String line = scanner.nextLine();  
 Matcher matcher = pattern.matcher(line);  
 // Замена "РА" на "РО" или "ра" на "ро", сохраняя регистр и следующую букву  
 String correctedLine = matcher.replaceAll(result ->  
 result.group().startsWith("Р") || result.group().startsWith("р") ? "ро" : "РО");  
 writer.println(correctedLine);  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Ошибка при работе с файлом: " + e.getMessage());  
 }  
 }  
}

Результат работы программы представлен на рисунке 1.

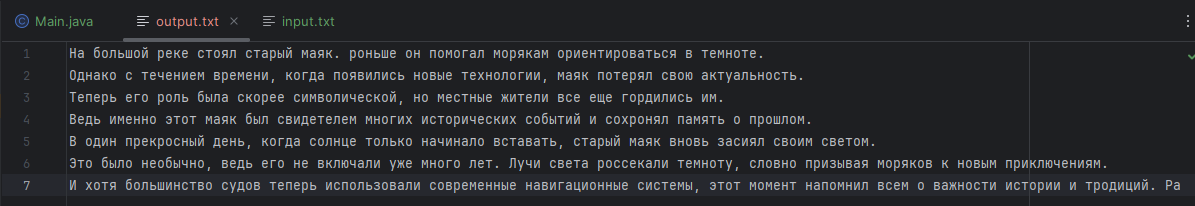


Рисунок 1 – Результат работы программы

**Задание 2:** в тексте слова заданной длины заменить указанной подстрокой, длина которой может не совпадать с длиной слова

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
 System.out.print("Введите длину слов для замены: ");  
 int wordLength = in.nextInt();  
 in.nextLine(); // Очистка буфера после чтения числа  
  
 System.out.print("Введите подстроку для замены: ");  
 String replacement = in.nextLine();  
  
 String sourceFilePath = "input.txt"; // Замените на ваш путь к исходному файлу  
 String outputFilePath = "output.txt"; // Замените на ваш путь к выходному файлу  
  
 // Создаем шаблон для поиска слов заданной длины  
 Pattern pattern = Pattern.compile("\\b\\w{" + wordLength + "}\\b", Pattern.UNICODE\_CHARACTER\_CLASS);  
  
 try (Scanner fileScanner = new Scanner(new File(sourceFilePath));  
 PrintWriter writer = new PrintWriter(outputFilePath)) {  
 while (fileScanner.hasNextLine()) {  
 String line = fileScanner.nextLine();  
 Matcher matcher = pattern.matcher(line);  
 String modifiedLine = matcher.replaceAll(replacement);  
 writer.println(modifiedLine);  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 }  
  
 in.close();  
 }  
}

Результат работы программы показан на рисунке 2.

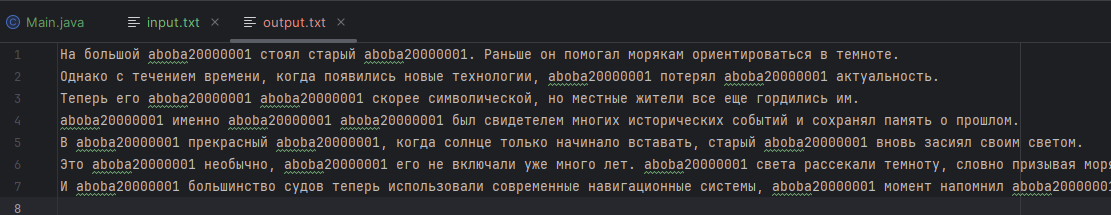


Рисунок 2 – Результат работы программы

**Задание 3:** в тексте найти и напечатать n символов (и их количество), встречающихся наиболее часто.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Scanner;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Scanner consoleScanner = new Scanner(System.in);  
 System.out.print("Введите количество символов для отображения: ");  
 int n = consoleScanner.nextInt();  
 consoleScanner.nextLine(); // Очистка буфера после чтения числа  
  
 String filePath = "input.txt";  
  
 Map<Character, Integer> charCounts = new HashMap<>();  
  
 try (Scanner fileScanner = new Scanner(new File(filePath))) {  
 while (fileScanner.hasNextLine()) {  
 String line = fileScanner.nextLine();  
 for (char c : line.toCharArray()) {  
 charCounts.put(c, charCounts.getOrDefault(c, 0) + 1);  
 }  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 return;  
 }  
  
 // Сортируем символы по количеству их встречаемости и выводим первые n  
 charCounts.entrySet().stream()  
 .sorted(Map.Entry.<Character, Integer>comparingByValue().reversed())  
 .limit(n)  
 .forEach(entry -> System.out.println("'" + entry.getKey() + "' встречается " + entry.getValue() + " раз(а)"));  
  
 consoleScanner.close();  
 }  
}

Результат работы программы представлен на рисунке 3.

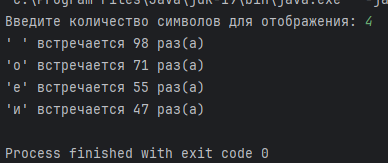
****

Рисунок 3 – Результат работы программы

**Задание 4:** найти, каких букв, гласных или согласных, больше в каждом предложении текста.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 private static final Pattern VOWELS\_PATTERN = Pattern.compile("[aeiouAEIOUаеёиоуыэюяАЕЁИОУЫЭЮЯ]");  
 private static final Pattern CONSONANTS\_PATTERN = Pattern.compile("[bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZбвгджзйклмнпрстфхцчшщБВГДЖЗЙКЛМНПРСТФХЦЧШЩ]");  
  
 public static void main(String[] args) {  
 String filePath = "input.txt"; // Замените на путь к вашему текстовому файлу  
  
 try (Scanner scanner = new Scanner(new File(filePath))) {  
 scanner.useDelimiter("[.!?]"); // Разделение текста на предложения  
  
 while (scanner.hasNext()) {  
 String sentence = scanner.next();  
 Matcher vowelsMatcher = VOWELS\_PATTERN.matcher(sentence);  
 Matcher consonantsMatcher = CONSONANTS\_PATTERN.matcher(sentence);  
  
 int vowelsCount = 0;  
 while (vowelsMatcher.find()) {  
 vowelsCount++;  
 }  
  
 int consonantsCount = 0;  
 while (consonantsMatcher.find()) {  
 consonantsCount++;  
 }  
  
 if (vowelsCount > consonantsCount) {  
 System.out.println("В предложении больше гласных: " + sentence.trim());  
 } else if (consonantsCount > vowelsCount) {  
 System.out.println("В предложении больше согласных: " + sentence.trim());  
 } else {  
 System.out.println("Гласных и согласных поровну: " + sentence.trim());  
 }  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 }  
 }  
}

Результат работы программы показан на рисунке 4.

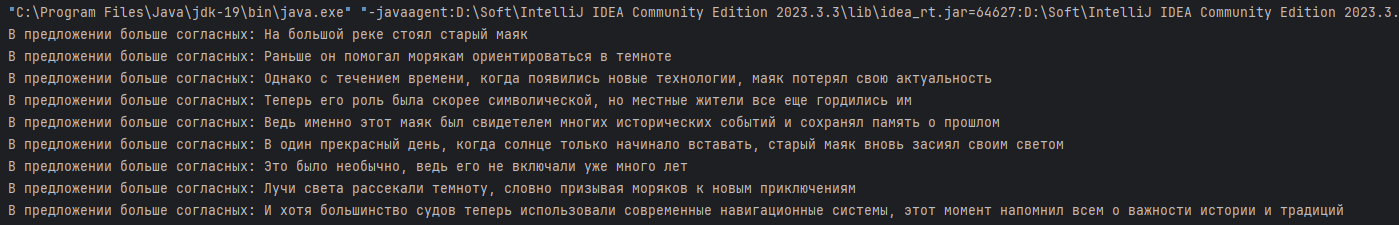


Рисунок 4 – Результат работы программы

**Задание 5:** найти такое слово в первом предложении, которого нет ни в одном из остальных предложений.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.HashSet;  
import java.util.Scanner;  
import java.util.Set;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 String filePath = "input.txt";  
 try (Scanner scanner = new Scanner(new File(filePath))) {  
 scanner.useDelimiter("[.!?]"); // Разделение текста на предложения  
  
 // Чтение и сохранение всех предложений  
 String firstSentence = scanner.hasNext() ? scanner.next().trim() : null;  
 StringBuilder restText = new StringBuilder();  
 while (scanner.hasNext()) {  
 restText.append(scanner.next());  
 }  
  
 if (firstSentence != null) {  
 // Находим уникальное слово в первом предложении  
 findUniqueWord(firstSentence, restText.toString());  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 }  
 }  
  
 private static void findUniqueWord(String firstSentence, String restText) {  
 Pattern wordPattern = Pattern.compile("\\b\\w+\\b", Pattern.UNICODE\_CHARACTER\_CLASS);  
 Matcher firstSentenceMatcher = wordPattern.matcher(firstSentence);  
 Set<String> wordsInRestText = new HashSet<>();  
 Matcher restTextMatcher = wordPattern.matcher(restText);  
  
 // Сохраняем все слова из оставшегося текста в Set для быстрого поиска  
 while (restTextMatcher.find()) {  
 wordsInRestText.add(restTextMatcher.group().toLowerCase());  
 }  
  
 // Ищем уникальное слово в первом предложении  
 while (firstSentenceMatcher.find()) {  
 String word = firstSentenceMatcher.group().toLowerCase();  
 if (!wordsInRestText.contains(word)) {  
 System.out.println("Уникальное слово в первом предложении: " + word);  
 return; // Выводим первое найденное уникальное слово и завершаем метод  
 }  
 }  
  
 System.out.println("Уникальных слов в первом предложении не найдено.");  
 }  
}

Результат работы программы представлен на рисунке 5.

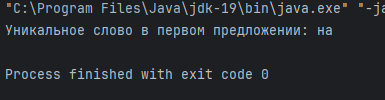


Рисунок 5 – Результат работы программы

**Задание 6:** во всех вопросительных предложениях текста найти и напечатать без повторений слова заданной длины.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.HashSet;  
import java.util.Scanner;  
import java.util.Set;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 String filePath = "input.txt"; // Замените на путь к вашему текстовому файлу  
 Scanner consoleScanner = new Scanner(System.in);  
 System.out.print("Введите длину слов для поиска: ");  
 int wordLength = consoleScanner.nextInt();  
  
 try (Scanner scanner = new Scanner(new File(filePath))) {  
 String text = scanner.useDelimiter("\\A").next(); // Читаем весь текст файла  
 Pattern questionPattern = Pattern.compile("[^.!?]\*\\?", Pattern.UNICODE\_CHARACTER\_CLASS);  
 Matcher questionMatcher = questionPattern.matcher(text);  
  
 Set<String> uniqueWords = new HashSet<>();  
 while (questionMatcher.find()) {  
 String question = questionMatcher.group();  
 Pattern wordPattern = Pattern.compile("\\b\\w{" + wordLength + "}\\b", Pattern.UNICODE\_CHARACTER\_CLASS);  
 Matcher wordMatcher = wordPattern.matcher(question);  
  
 while (wordMatcher.find()) {  
 uniqueWords.add(wordMatcher.group().toLowerCase()); // Добавляем слова в нижнем регистре для избежания дубликатов  
 }  
 }  
  
 System.out.println("Уникальные слова длиной " + wordLength + " в вопросительных предложениях:");  
 uniqueWords.forEach(System.out::println);  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 }  
  
 consoleScanner.close();  
 }  
}

Результат работы программы представлен на рисунке 6.

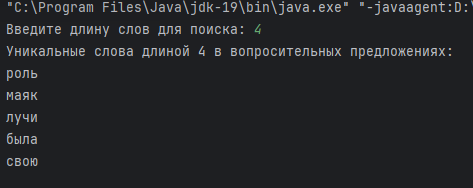


Рисунок 6 – Результат работы программы

**Задание 7:** все слова текста рассортировать в порядке убывания их длин, при этом все слова одинаковой длины рассортировать в порядке возрастания в них количества гласных букв.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.ArrayList;  
import java.util.Comparator;  
import java.util.List;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 private static int countVowels(String word) {  
 int vowels = 0;  
 for (char c : word.toLowerCase().toCharArray()) {  
 if ("aeiouаеёиоуыэюя".indexOf(c) != -1) {  
 vowels++;  
 }  
 }  
 return vowels;  
 }  
  
 public static void main(String[] args) {  
 String filePath = "input.txt";  
  
 List<String> words = new ArrayList<>();  
 Pattern wordPattern = Pattern.compile("\\b\\w+\\b", Pattern.UNICODE\_CHARACTER\_CLASS);  
  
 try (Scanner scanner = new Scanner(new File(filePath))) {  
 while (scanner.hasNextLine()) {  
 String line = scanner.nextLine();  
 Matcher matcher = wordPattern.matcher(line);  
 while (matcher.find()) {  
 words.add(matcher.group());  
 }  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 return;  
 }  
  
 words.sort(Comparator.comparingInt(Main::countVowels)  
 .thenComparingInt(String::length).reversed()  
 .thenComparing(Comparator.naturalOrder()));  
  
 words.forEach(System.out::println);  
 }  
}

Результат работы программы представлен на рисунке 7.

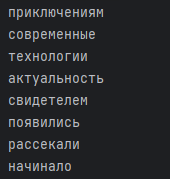


Рисунок 7 – Результат работы программы

**Задание 8:** в тексте исключить подстроку максимальной длины, начинающуюся и заканчивающуюся заданными символами.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Scanner consoleScanner = new Scanner(System.in);  
 System.out.print("Введите начальный символ подстроки: ");  
 char startSymbol = consoleScanner.next().charAt(0);  
 System.out.print("Введите конечный символ подстроки: ");  
 char endSymbol = consoleScanner.next().charAt(0);  
  
 String filePath = "input.txt";  
  
 StringBuilder text = new StringBuilder();  
 try (Scanner fileScanner = new Scanner(new File(filePath))) {  
 fileScanner.useDelimiter("\\A"); // Читаем весь файл как одну строку  
 if (fileScanner.hasNext()) {  
 text.append(fileScanner.next());  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 consoleScanner.close();  
 return;  
 }  
  
 String regex = Pattern.quote(String.valueOf(startSymbol)) + ".\*?" + Pattern.quote(String.valueOf(endSymbol));  
 Pattern pattern = Pattern.compile(regex);  
 Matcher matcher = pattern.matcher(text);  
  
 String longestSubstring = "";  
 int longestStart = -1;  
 int longestEnd = -1;  
  
 String max = "";  
 while (matcher.find()) {  
 String currentMatch = matcher.group();  
 if (currentMatch.length() > longestSubstring.length()) {  
 max = currentMatch;  
 longestSubstring = currentMatch;  
 longestStart = matcher.start();  
 longestEnd = matcher.end();  
 }  
 }  
  
 System.out.println("Удалена подстрока: " + max);  
 if (longestStart != -1 && longestEnd != -1) {  
 text.delete(longestStart, longestEnd);  
 }  
  
 System.out.println(text.toString());  
 consoleScanner.close();  
 }  
}

Результат работы программы представлен на рисунке 8.

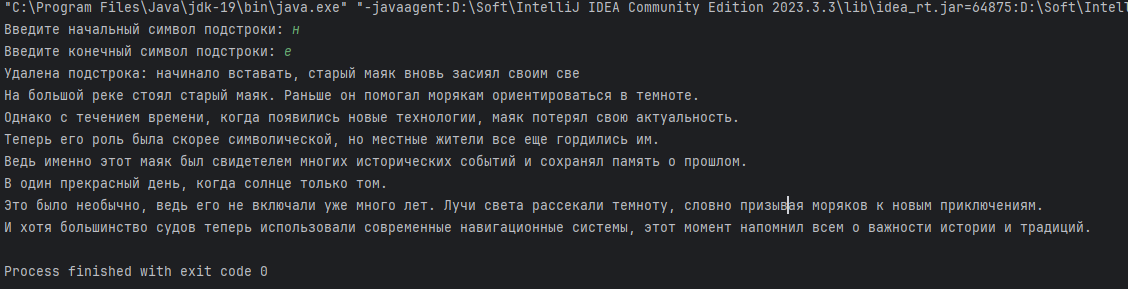


Рисунок 8 – Результат работы программы

**Вывод:** была освоена работа со строками и регулярными выражениями в Java.