|  |  |
| --- | --- |
| **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 №** | 5 |

**Название:**

Исключения и файлы

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

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

Москва, 2023

**Вариант 3, задание 7:** В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. При этом могут рассматриваться два варианта:

\* • каждая строка состоит из одного слова;

\* • каждая строка состоит из нескольких слов.

\* Имена входного и выходного файлов, а также абсолютный путь к ним могут быть введены как параметры командной строки или храниться в файле

\* В каждом слове стихотворения Николая Заболоцкого заменить первую букву слова на прописную.

Код

|  |
| --- |
| import java.io.BufferedReader;  import java.io.FileReader;  import java.io.IOException;  import java.io.PrintWriter;  public class Main {  public static void main(String[] args) throws IOException {  // Проверяем, были ли переданы аргументы командной строки  if (args.length != 2) {  System.out.println("Использование: java Main inputFile outputFile");  return;  }  // Открываем файлы для чтения и записи  BufferedReader input = new BufferedReader(new FileReader("C:\\University\\Java-BMSTU\\Laboratory5\\" + args[0]));  PrintWriter output = new PrintWriter("C:\\University\\Java-BMSTU\\Laboratory5\\" + args[1]);  // Читаем входные данные построчно  String line;  while ((line = input.readLine()) != null) {  // Разбиваем строку на слова  String[] words = line.split(" ");  // Заменяем первую букву каждого слова на прописную  for (int i = 0; i < words.length; i++) {  String word = words[i];  if (!word.isEmpty()) {  char firstChar = Character.toUpperCase(word.charAt(0));  words[i] = firstChar + word.substring(1);  }  }  // Объединяем слова обратно в строку  String result = String.join(" ", words);  // Записываем полученный результат в выходной файл  output.println(result);  }  // Закрываем файлы  input.close();  output.close();  }  } |

**Вариант 3, задание 8:** В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия.

\* При этом могут рассматриваться два варианта:

\* • каждая строка состоит из одного слова;

\* • каждая строка состоит из нескольких слов.

\* Имена входного и выходного файлов, а также абсолютный путь к ним могут быть введены как параметры командной строки или храниться в файле

\* Определить частоту повторяемости букв и слов в стихотворении Александра Пушкина.

Код

|  |
| --- |
| import java.io.BufferedReader;  import java.io.FileReader;  import java.io.IOException;  import java.io.PrintWriter;  import java.util.\*;  public class Main {  public static void main(String[] args) throws IOException {  // Проверяем, были ли переданы аргументы командной строки  if (args.length != 2) {  System.out.println("Использование: java Main inputFile outputFile");  return;  }  // Открываем файлы для чтения и записи  BufferedReader input = new BufferedReader(new FileReader("C:\\University\\Java-BMSTU\\Laboratory5\\" + args[0]));  PrintWriter output = new PrintWriter("C:\\University\\Java-BMSTU\\Laboratory5\\" + args[1]);  Map<String, Integer> wordFrequency = new HashMap<>();  Map<Character, Integer> letterFrequency = new HashMap<>();  // Читаем входные данные построчно  String line;  while ((line = input.readLine()) != null) {  // Разбиваем строку на слова  String[] words = line.split(" ");  for (int i = 0; i < words.length; i++) {  String word = words[i];  if (!word.isEmpty()) {  wordFrequency.put(word, wordFrequency.getOrDefault(word, 0) + 1);  for (char c : word.toCharArray()) {  letterFrequency.put(c, letterFrequency.getOrDefault(c, 0) + 1);  }  }  }  }  for (Map.Entry<String, Integer> entry : wordFrequency.entrySet()) {  output.println(entry.getKey() + " : " + entry.getValue());  }  for (Map.Entry<Character, Integer> entry : letterFrequency.entrySet()) {  output.println(entry.getKey() + " : " + entry.getValue());  }  // Закрываем файлы  input.close();  output.close();  }  } |

**Вариант 4, задание 7:** При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File. Прочитать текст Java-программы и удалить из него все “лишние” пробелы и табуляции, оставив только необходимые для разделения операторов.

Код

|  |
| --- |
| import java.io.\*;  public class Main {  public static void main(String[] args) {  try {  // Создаем новую директорию и файл  File dir = new File("C:\\University\\Java-BMSTU\\Laboratory5\\" + "output");  dir.mkdir();  File file = new File(dir, "output.txt");  file.createNewFile();  // Читаем входной файл  BufferedReader reader = new BufferedReader(new FileReader("C:\\University\\Java-BMSTU\\Laboratory5\\Variant4\_7.java"));  String line = null;  StringBuilder sb = new StringBuilder();  while ((line = reader.readLine()) != null) {  sb.append(line.trim() + "\n"); // Удаляем лишние пробелы и табуляции  }  reader.close();  // Записываем результат в выходной файл  FileWriter writer = new FileWriter(file);  writer.write(sb.toString());  writer.close();  System.out.println("Completely done!");  } catch (IOException e) {  System.out.println("Error: " + e.getMessage());  }  }  } |

**Вариант 4, задание 8:** При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File. Из текста Java-программы удалить все виды комментариев.

Код

|  |
| --- |
| import java.io.\*;  import java.util.regex.\*;  public class RemoveComments {  public static void main(String[] args) throws IOException {  // Создаем новый файл для результатов  File dir = new File("C:\\University\\Java-BMSTU\\Laboratory5\\" + "output1");  dir.mkdir();  File file = new File(dir, "output.txt");  file.createNewFile();  // Читаем исходный файл  BufferedReader reader = new BufferedReader(new FileReader("C:\\University\\Java-BMSTU\\Laboratory5\\Variant4\_8.java"));  String line = null;  StringBuilder sb = new StringBuilder();  boolean isMultilineComment = false;  while ((line = reader.readLine()) != null) {  if (!isMultilineComment) {  // Удаляем однострочные комментарии  int commentIndex = line.indexOf("//");  if (commentIndex != -1) {  line = line.substring(0, commentIndex);  }  // Проверяем, начинается ли многострочный комментарий  int openCommentIndex = line.indexOf("/\*");  if (openCommentIndex != -1) {  isMultilineComment = true;  line = line.substring(0, openCommentIndex);  }  } else {  // Закрываем многострочный комментарий  int closeCommentIndex = line.indexOf("\*/");  if (closeCommentIndex != -1) {  isMultilineComment = false;  line = line.substring(closeCommentIndex + 2);  } else {  line = "";  }  }  sb.append(line + "\n");  }  FileWriter writer = new FileWriter(file);  writer.write(sb.toString());  writer.close();  reader.close();  }  } |