|  |  |
| --- | --- |
| 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, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ и системы  
 управления**

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

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. Файлы

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

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

**Цель работы** — ознакомление с механизмами работы с исключениями и файлами в языке Java.

1. Выполнить задания на основе варианта 1 лабораторной работы 3, контролируя состояние потоков ввода/вывода. При возникновении ошибок, связанных с корректностью выполнения математических операций, генерировать и обрабатывать исключительные ситуации. Предусмотреть обработку исключений, возникающих при нехватке памяти, отсутствии требуемой записи (объекта) в файле, недопустимом значении поля и т.д.

Часть кода задания приведена в листингах 1 и 2, результат выполнения – на рисунках 1 и 2.

Листинг 1 – Исключение в классе Вектор при нормализации

|  |
| --- |
| public Vector normalize() throws Exception {  int mag = 0;  try {  mag = (int) magnitude();  return new Vector((int) (x / mag), (int) (y / mag), (int) (z / mag));  } catch (Exception e) {  System.out.printf("%s\n", e.toString());  throw e;  }  } |



Рисунок 1 – Вывод исключения

Листинг 2 – Исключение при вычитании матриц

|  |
| --- |
| public Matrix substract(Matrix m) throws Exception {  int[][] arr = new int[this.n][this.n];  try {  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  arr[i][j] = this.arr[i][j] - m.arr[i][j];  }  }  } catch (Exception e) {  System.out.printf("%s\n", e.toString());  throw e;  }  return new Matrix(arr);  } |



Рисунок 2 – Выход за границы при вычитании

1. Выполнить задания из варианта 2 лабораторной работы 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

Часть кода задания приведена в листингах 3 и 4, результат выполнения – на рисунках 3 и 4.

Листинг 3 – Исключение некорректного телефона

|  |
| --- |
| public class InvalidPhoneException extends Exception {  public InvalidPhoneException(String message) {  super(message);  System.out.println("phone should have 11 numbers");  }  } |



Рисунок 3 – Пример вывода исключения

Листинг 4 – Обработка исключения некорректного телефона

|  |
| --- |
| public static void main(String[] args) {  Abiturient abiturient = new Abiturient(1, "Ivanov", "Ivan", "Ivanovich", "123 Main St", "1234567",  new float[] { 3, 4, 5, 1 });  try {  abiturient.setPhoneNumber("8800555353");  } catch (Exception e) {  System.out.println("error");  }  }  } |



Рисунок 4 – Вывод исключения

1. 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.

* 1. В каждой строке найти слова, начинающиеся с гласной буквы.

Часть кода задания приведена в листинге 5, результат выполнения – на рисунке 5.

Листинг 5 – Нахождение слов, начинающихся с гласных букв

|  |
| --- |
| public static ArrayList<String> startsWithVowel(ArrayList<String> lines) {  ArrayList<String> vowelWords = new ArrayList<>();  for (String s : lines) {  String[] words = s.split(" ");  for (String w : words) {  if (w.charAt(0) == 'a' || w.charAt(0) == 'e'  || w.charAt(0) == 'i' || w.charAt(0) == 'o'  || w.charAt(0) == 'u') {  vowelWords.add(w);  }  }  }  return vowelWords;  } |

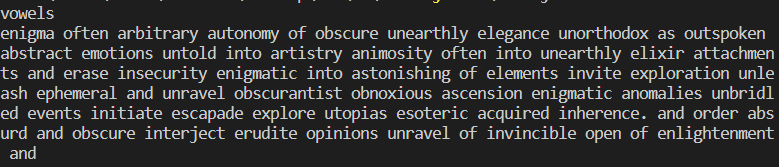


Рисунок 5 – Слова, начинающиеся на гласную

* 1. Найти и вывести слова текста, для которых последняя буква одного слова совпадает с первой буквой следующего слова.

Часть кода задания приведена в листинге 6, результат выполнения – на рисунке 6.

Листинг 6 – Нахождение слов по требуемому условию

|  |
| --- |
| public static ArrayList<String> lastAsFirst(ArrayList<String> lines) {  ArrayList<String> lastFirst = new ArrayList<>();  for (String s : lines) {  String[] words = s.split(" ");  for (int i = 0; i < words.length - 1; i++) {  if (words[i].charAt(words[i].length() - 1) == words[i + 1].charAt(0)) {  lastFirst.add(words[i] + " " + words[i + 1]);  }  }  }  return lastFirst;} |

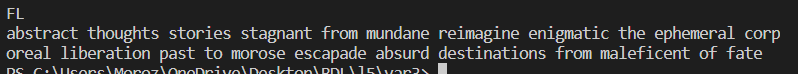


Рисунок 6 – Слова по требуемому условию

1. При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File.
   1. Прочитать текст Java-программы и в каждом слове длиннее двух символов все строчные символы заменить прописными.

Часть кода задания приведена в листинге 7, результат выполнения – на рисунке 7.

Листинг 7 – Замена строчных символов на прописные

|  |
| --- |
| public static String replaceUpper(ArrayList<String> lines) {  String text = "";  for (String line : lines) {  String[] words = line.split(" ");  for (int i = 0; i < words.length; i++) {  if (words[i].length() > 2) {  words[i] = words[i].toUpperCase();  }  }  text += String.join(" ", words) + "\n";  }  return text;  } |

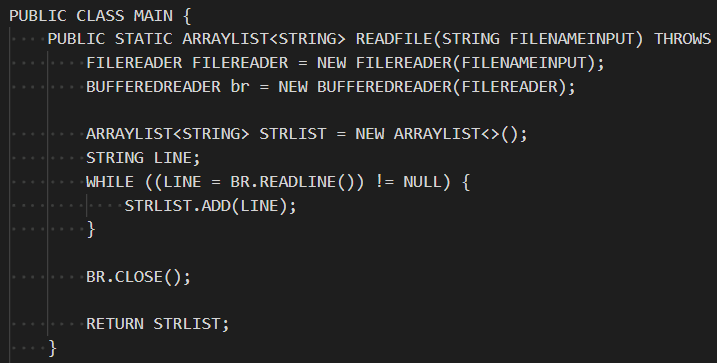


Рисунок 7 – Часть программы после замены символов

* 1. В файле, содержащем фамилии студентов и их оценки, записать прописными буквами фамилии тех студентов, которые имеют средний балл более «7».

Часть кода задания приведена в листинге 8, результат выполнения – на рисунке 8.

Листинг 8 – Поиск студентов со средним баллом больше «7»

|  |
| --- |
| public static ArrayList<Integer> findStudents(ArrayList<String> lines) {  ArrayList<Integer> indexes = new ArrayList<Integer>();  int idx = 0;  for (String line : lines) {  String[] data = line.split(" ");  double sum = 0;  for (int i = 1; i < data.length; i++) {  double mark = Double.parseDouble(data[i]);  sum += mark;  }  double average = sum / (data.length - 1);  if (average > 7.0) {  indexes.add(idx);  }  idx++;  }  return indexes;  } |

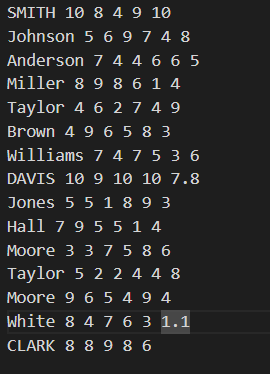


Рисунок 8 – Отредактированный файл

Полный код заданий размещен в репозитории по ссылке – https://github.com/moroz-matros/BDL.

**Вывод** — в ходе работы были получены навыки работы с исключениями и файлами в языке Java.