|  |  |
| --- | --- |
| 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 **09.04.01/05 Современные интеллектуальные**

**программно-аппаратные комплексы**

**ОТЧЕТ**

**по лабораторной работе №6**

|  |  |
| --- | --- |
| Группа | ИУ6-23М |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | И.И. Иванов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |
| Преподаватель |  |  | П.В. Степанов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |

*Москва, 2023 г.*

**Условие**

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.

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

3.9. Входной файл содержит совокупность строк. Строка файла содержит строку квадратной матрицы. Ввести матрицу в двумерный массив (размер матрицы найти). Вывести исходную матрицу и результат ее транспонирования.  
3.10. Входной файл хранит квадратную матрицу по принципу: строка представляет собой число. Определить размерность. Построить 2-мерный массив, содержащий матрицу. Вывести исходную матрицу и результат ее поворота на 90 градусов по часовой стрелке.

4.9. Прочитать строки из файла и поменять местами первое и последнее слова в каждой строке.

4.10. Ввести из текстового файла, связанного с входным потоком, последовательность строк. Выбрать и сохранить m последних слов в каждой из последних n строк

**Програмная реализация**

1.9.

|  |
| --- |
| public class Main {  public static void main(String[] args) {  Set set1 = new Set(1, 2, 3);  Set set2 = new Set(2, 3, 4, 5);  System.out.println("Intersection: " + set1.intersect(set2));  System.out.println("Union: " + set1.unify(set2));  }  } |

1.10.

|  |
| --- |
| public class Main {  public static void main(String[] args) {  int n = 10;  double[] I = {1.2, 2.1, 3.7, 4.5, 5.6, 6.8, 7.4, 8.2, 9.0, 10.1};  double[] U = {2.1, 3.4, 4.9, 6.0, 7.5, 9.1, 10.2, 11.8, 12.5, 14.3};  double avgI = 0;  double avgU = 0;  for (int i = 0; i < n; i++) {  avgI += I[i];  avgU += U[i];  }  avgI /= n;  avgU /= n;  double sumProd = 0;  double sumSqrU = 0;  for (int i = 0; i < n; i++) {  sumProd += (I[i] - avgI) \* (U[i] - avgU);  sumSqrU += Math.pow(U[i] - avgU, 2);  }  double R = sumProd / sumSqrU;  System.out.println("R = " + R);  }  } |

3.9.

|  |
| --- |
| public class Main {  public static void main(String[] args) {  int[][] matrix = {  {1, 2, 2, 4},  {5, 2, 2, 8},  {9, 10, 11, 12},  {13, 14, 15, 16}  };  int maxCount = 0;  int maxI = -1;  int maxJ = -1;  int maxSizeI = -1;  int maxSizeJ = -1;  for (int i = 0; i < matrix.length; i++) {  for (int j = 0; j < matrix[i].length; j++) {  for (int sizeI = 1; sizeI <= matrix.length - i; sizeI++) {  for (int sizeJ = 1; sizeJ <= matrix[i].length - j; sizeJ++) {  int count = countEqualElements(matrix, i, j, sizeI, sizeJ);  if (count > maxCount) {  maxCount = count;  maxI = i;  maxJ = j;  maxSizeI = sizeI;  maxSizeJ = sizeJ;  }  }  }  }  }  if (maxCount > 0) {  System.out.println("Max count: " + maxCount);  System.out.println("Submatrix:");  for (int i = maxI; i < maxI + maxSizeI; i++) {  for (int j = maxJ; j < maxJ + maxSizeJ; j++) {  System.out.print(matrix[i][j] + " ");  }  System.out.println();  }  } else {  System.out.println("No equal elements found");  }  }  private static int countEqualElements(int[][] matrix, int i, int j, int sizeI, int sizeJ) {  int count = 0;  int element = matrix[i][j];  for (int ii = i; ii < i + sizeI; ii++) {  for (int jj = j; jj < j + sizeJ; jj++) {  if (matrix[ii][jj] == element) {  count++;  }  }  }  return count;  }  } |

3.10.

|  |
| --- |
| public class Main {  public static void main(String[] args) {  int[] positions = {10, 15, 20, 25}; // позиции  int[] speeds = {2, 3, 5, 4}; // скорости  int count = 0;  for (int i = 0; i < positions.length; i++) {  for (int j = i + 1; j < positions.length; j++) {  // если один автомобиль догоняет другой сзади  if (positions[i] < positions[j] && speeds[i] > speeds[j]) {  count++;  } else if (positions[j] < positions[i] && speeds[j] > speeds[i]) {  count++;  }  }  }  System.out.println("Количество обгонов: " + count);  }  } |