|  |  |
| --- | --- |
| 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 №** | 2 |

**Название:**

Арифметические операции

**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 арифметическими операциями на языке Java.

1. В приведенных ниже заданиях необходимо вывести внизу фамилию разработчика, дату и время получения задания, а также дату и время сдачи задания. Для получения последней даты и времени следует использовать класс Date.
   1. Ввести n строк с консоли. Вывести на консоль те строки, длина которых меньше (больше) средней, а также длину.

Часть кода задания приведена в листинге 1, результат выполнения – на рисунке 1.

Листинг 1 – Строки, длина которых меньше (больше) средней

|  |
| --- |
| for (int i = 0; i < n; i++) {  sum += arr[i].length();  }  int avg = sum / n;  System.out.printf("The length is less than average %d:\n", avg);  for (int i = 0; i < n; i++) {  if (arr[i].length() < avg) {  System.out.printf("%s with length %d\n", arr[i], arr[i].length());  }  }  System.out.printf("The length is greater than average %d:\n", avg);  for (int i = 0; i < n; i++) {  if (arr[i].length() > avg) {  System.out.printf("%s with length %d\n", arr[i], arr[i].length());  }  } |

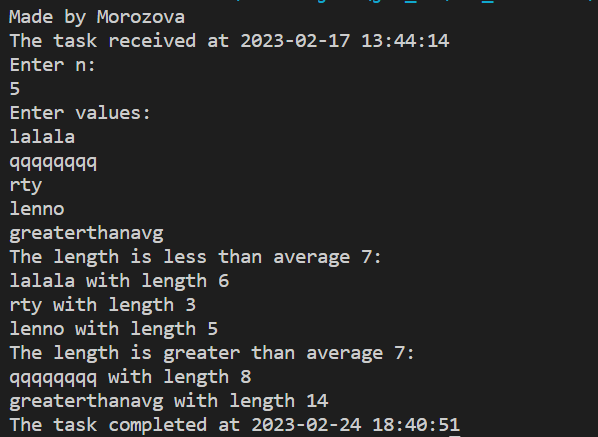


Рисунок 1 – Строки, длина которых меньше (больше) средней

* 1. Ввести n слов с консоли. Найти слово, в котором число различных символов минимально. Если таких слов несколько, найти первое из них.

Часть кода задания приведена в листинге 2, результат выполнения – на рисунке 2.

Листинг 2 – Первое слово с минимальным числом различных символов

|  |
| --- |
| long minimum = arr[0].chars().distinct().count();  int minimumIdx = 0;  for (int i = 1; i < n; i++) {  long lenDiff = arr[i].chars().distinct().count();  if (lenDiff < minimum) {  minimum = lenDiff;  minimumIdx = i;  }  } |

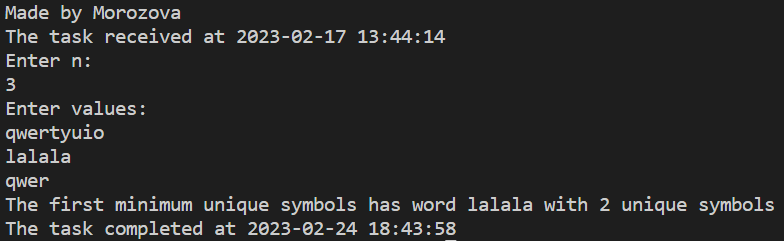


Рисунок 2 – Первое слово с минимальным числом различных символов

1. Ввести с консоли n – размерность матрицы a[n][n]. Задать значения элементов матрицы в интервале значений от -n до n с помощью датчика случайных чисел.
   1. Найти и вывести наибольшее число возрастающих (убывающих) элементов матрицы, идущих подряд.

Часть кода задания приведена в листинге 3, результат выполнения – на рисунке 3.

Листинг 3 – Наибольшее число возрастающих (убывающих) элементов матрицы, идущих подряд

|  |
| --- |
| for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  if (arr[i][j] < prev) {  decNums[0] += 1;  manageMaximum(incNums);  } else if (arr[i][j] > prev) {  incNums[0] += 1;  manageMaximum(decNums);  } else {  manageMaximum(incNums);  manageMaximum(decNums);  }  prev = arr[i][j];  }  } |

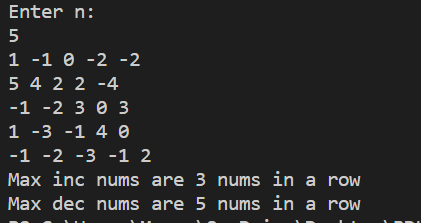


Рисунок 3 – Наибольшее число возрастающих (убывающих) элементов матрицы, идущих подряд

* 1. Найти сумму элементов матрицы, расположенных между первым и вторым положительными элементами каждой строки.

Часть кода задания приведена в листинге 4, результат выполнения – на рисунке 4.

Листинг 4 – Сумма элементов матрицы, расположенных между первым и вторым положительными элементами каждой строки

|  |
| --- |
| for (int i = 0; i < n; i++) {  boolean startedToSum = false;  boolean alreadySummed = false;  for (int j = 0; j < n; j++) {  if (arr[i][j] >= 0) {  // found first positive  if (!startedToSum && !alreadySummed) {  startedToSum = true;  continue;  }  // found second positive, cause already started to sum  if (startedToSum) {  startedToSum = false;  alreadySummed = true;  }  } else if (startedToSum) {  sums[i] += arr[i][j];  }  }  // didn't find second positive  if (!alreadySummed && startedToSum) {  sums[i] = 0;  }  } |

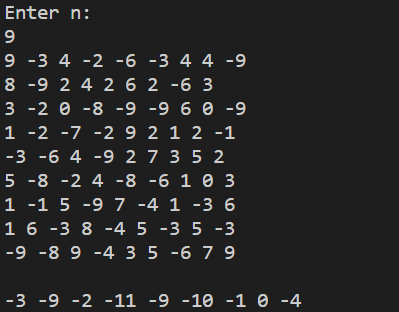


Рисунок 4 – Сумма элементов матрицы, расположенных между первым и вторым положительными элементами каждой строки

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

**Вывод** — в ходе работы были получены навыки работы с арифметическими операциями в языке Java.