Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

Центр программной инженерии

ОТЧЕТ

к лабораторной работе №3 по дисциплине:

|  |
| --- |
| «Объектно-ориентированное программирование» |
| Работа с потоками данных и обработка исключений |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСТб-19-2 |  |  |  | Карбушев И.В. |
|  | шифр группы |  | подпись |  | Фамилия ИО |
| Проверил: | доцент |  |  |  | Маланова Т.В. |
|  | должность |  | подпись |  | Фамилия ИО |

Иркутск. 2020 г.

Оглавление

[Оглавление 2](#_Toc55037499)

[1 Постановка задачи 3](#_Toc55037500)

[2 Описание структуры пользовательского меню 4](#_Toc55037501)

[3 Проектирование классов 5](#_Toc55037502)

[4 Описание методов классов и спецификация локальных переменных 6](#_Toc55037503)

[5 Таблицы тестов 11](#_Toc55037504)

[6 Результаты тестирования 16](#_Toc55037505)

[7 Код программы 22](#_Toc55037506)

[Список использованных источников 30](#_Toc55037507)

1 Постановка задачи

1. Применение на практике объектно-ориентированного подхода для орга-низации системы ввода/вывода.

2. Получение практического опыта описания и реализации классов, созда-ния объектов, вызова методов, а также навыков работы с потоками дан-ных и обработки исключений на базе платформы Java.

Вариант №7

a. Отсортируйте слова в массиве по третьей букве, если слово меньше трех букв, то по последней.

b. Определите, входит ли строка, указанная в командной строке, при запуске программы, в предложение.

c. Удалите в предложении каждое третье слово.

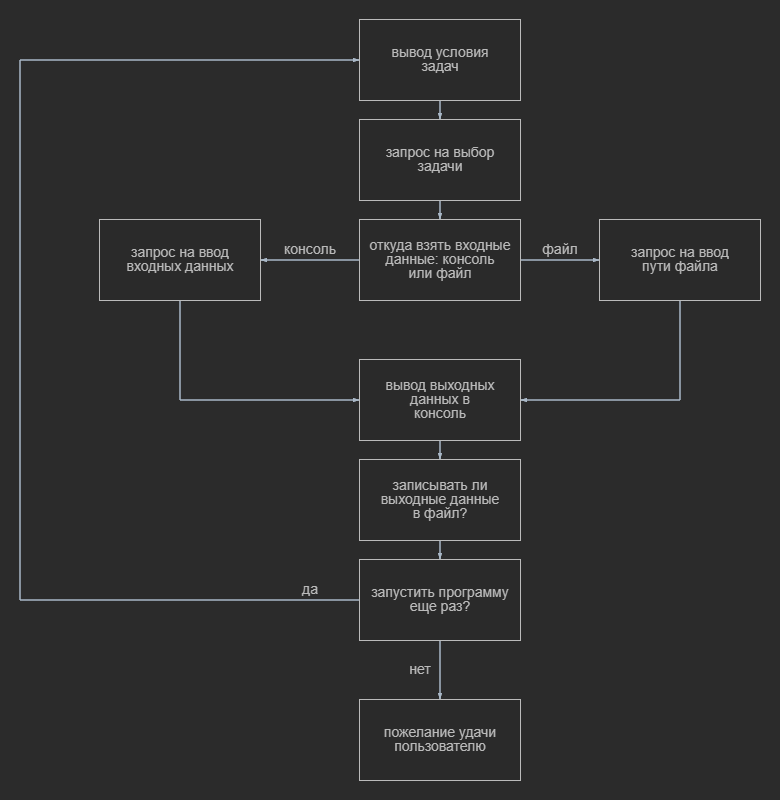
2 Описание структуры пользовательского меню

Рисунок 1 – Описание структуры пользовательского меню

3 Проектирование классов

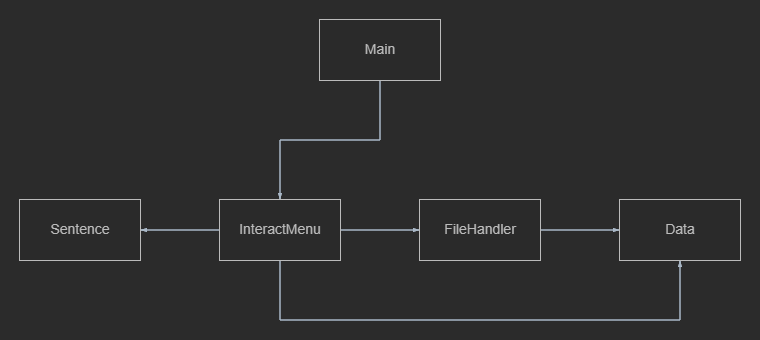


Рисунок 2 – Проектирование классов

4 Описание методов классов и спецификация локальных переменных

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных | |
| **Class Main** | | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner | |
| **Public static void main()** | | | | |
| 2 | reply | Ответ пользователя на повторный запуск программы | String | |
| 3 | number | Номер выбранного задания | int | |
| 4 | data | Входные данные для задания | Data | |
| 5 | res | Выходные данные с задания | String | |
| **Class Sentence** | | | | |
| 1 | stringArray | Поле класса для работы с методами | String[] | |
| **Private void swapValues(int ind1, int ind2)** | | | | |
| 2 | temp | Временная переменная для обмена местами значений массива | String | |
| 3 | ind1, ind2 | Индексы массива, значения которых нужно поменять местами | int | |
| **Public void sort()** | | | | |
| 1 | i,j | Переменные в циклах | int | |
| 2 | str2, str1 | Текущий элемент массива и первый не отсортированный элемент массива | String | |
| **Public Boolean ifContains(String subStr)** | | | | |
| 1 | i | Переменная в цикле | int | |
| 2 | count | Переменная-счетчик для проверки, содержится ли подстрока subSentence в строке sentence | int | |
| 3 | sent | Строка из поля класса stringArray для удобной реализации метода | String | |
| 4 | sentence | Глобальная переменная класса для удобной работы с методами | String[] | |
| **Public void delWords(int number)** | | | | |
| 1 | count | Счетчик слов, чтобы удалять каждое 3 слово в строке | | int |
| 2 | number | Номер удаляемых слов в предложении | | int |
| **@Override**  **Public String toString()** | | | | |
| 1 | out | Массив, переведенный в строку | | StringBuilder |
| 2 | s | Элементы массива | | String |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Class InteractMenu** | | | |
| Public static int taskNumber() | | | |
| 1 | num | Номер задания | Ввод – String  Вывод - int |
| **Public static String ifNumber()** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | number | Число для проверки на корректность | Ввод – int  Вывод - String |
| **Public static String readWay()** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | reply | Ответ на вопрос, откуда прочитать данные | String |
| 3 | - | Возвращает “console” или “file” | String |
| **Public static Data getData(int number)** | | | |
| 1 | way | Каким путем считать данные | String |
| 2 | data | Хранение входных данных | Data |
| 3 | fileData | Данные с файла | String |
| **Public static Data inputData(int num)** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | input | Первое предложение заданий | String |
| 3 | subString | Подстрока 2 задачи | String |
| 4 | temp | Номер удаляемого слова в строке | String |
| 5 | number | Номер удаляемого слова в предложении | int |
| 6 | data | Выходные данные задач | Data |
| **Public static String solve(Data data, int number)** | | | |
| 1 | res | Результат решения задач | String |
| 2 | sentence | Первое предложение для всех 3-х задач | String[] |
| 3 | subString | Подстрока для 2ой задачи | String |
| 4 | count | Номер удаляемого слова для 3ей задачи | int |
| **Public static Sentence solveFirst(Sting[] sentence)** | | | |
| 1 | sent | Первое предложение в виде объекта класса Sentence | Sentence |
| **Public static Sentence solveSecond(Sting[] sentence, String subString)** | | | |
| 1 | sent | Первое предложение в виде объекта класса Sentence | Sentence |
| 2 | ans | Ответ на 2ую задачу | String |
| **Public static Sentence solveThird(String[] sentence, int number)** | | | |
| 1 | sent | Первое предложение в виде объекта класса Sentence | Sentence |
| **Class FileHandler** | | | |
| **Public static String readFile()** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | flag | Показывает, введен ли верно путь файла | Boolean |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 3 | res | Данные из файла | StringBuilder |
| 4 | path | Путь файла | String |
| 5 | fis | Поток чтения из файла | FileInputStream |
| **Public static Data fileInformationSplit(String fileData, int taskNumber)** | | | |
| 1 | arrData | Данные из файла, разделенные по специальному символу “;” | String[] |
| 2 | data | Данные из файла для задания | Data |
| 3 | temp | Второе предложение из файла | String |
| 4 | number | Номер удаляемого слова в предложении | int |
| **Public static void writeFile(String data)** | | | |
| 1 | in | Объект класса Scanner для ввода данных | Scanner |
| 2 | path | Путь файла для записи данных | String |
| 3 | fos | Поток записи данных | FIleOutputStream |
| 4 | pw | Посимвольная запись данных в файл | PrintWriter |
| **Class Data** | | | |
| 1 | sentence | Поле для 1ого предложения в задачах | String |
| 2 | subString | Поле для подстроки во 2 задаче | String |
| 3 | number | Поле для номера удаляемого слова в предложении в 3 задаче | int |

**Class Sentence:**

1. Public Sentence() – конструктор без любых входных данных
2. Public Sentence(String[] stringArray) – конструктор с входным массивом
3. Public Sentence(Sentence sent) – конструктор копирования с входным объектом этого же класса
4. Private void swapValues(int ind1, int ind2) – метод для того, чтобы менять значения массива местами
5. Public void sort() – главный метод класса, отвечает за сортировку массива по правилам из задания 1
6. Public boolean ifContains(String subStr) – определяет, содержится ли подстрока subStr в строке sentence
7. Public void delWords(int number) – удаляет каждое number-е слово в строке
8. @Override Public String toString() – геттер для переменной sentence
9. Public String[] getStringArray() – геттер для поля класса

**Class InteractMenu:**

1. Public static void start() – вывод списка заданий
2. Public static int taskNumber() – запрос на корректный ввод номера задания
3. Public static String ifNumber() – проверяет, введено ли число
4. Public static String readWay() – запрашивает путь ввода: консоль или файл
5. Public static Boolean ifWrite() – записывать ли выходные данные в файл
6. Public static void outData(String data) – записывает данные в файл, если нужно
7. Public static Data getData(int number) – получает и преобразовывает входные данные в нужный формат
8. Public static Data inputData(int num) – запрос на ввод входных данных
9. Public static String solve(Data data, int number) – решение какой-то задачи из трех
10. Public static String solve(Data data, int number) – вызов решения какой-то задачи из трех
11. Public static String solveFirst(String[] sentence) – решение первой задачи
12. Public static String solveSecond(String[] sentence, String, subString) – решение второй задачи
13. Public static String solveThird(String[] sentence, int number) – решение третьей задачи

**Class FileHandler:**

1. Public static String readFile() – побайтовое чтение данных из файла
2. Public static Data fileInformationSplit(String fileData, int taskNumber) – разделение данных из файла по символу «;» и приведение в нужный формат – объект класса Data
3. Public static void writeFile(String data) – посимвольная запись файлов в файл

**Class Data:**

1. Public Data(String sentence) – конструктор для первой задачи
2. Public Data(String sentence, String subString) – конструктор для второй задачи
3. Public Data(String sentence, int number) – конструктор для третьей задачи
4. Public String getSentence() – геттер для поля sentence
5. Public String getSubString() – геттер для поля subString
6. Public int getNumber() – геттер для поля number

5 Таблицы тестов

Таблица 2 – первая задача

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Данные на выходе** |
| 1 | 1  0  Раз два три четыре пять шесть  1  D:\test.txt  нет | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла)  Введите предложение  Результат решения первой задачи :  раз два три четыре пять шесть  Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Удачи |
| 2 | 1  1  D:\first.txt  1  D:\test.txt  net | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла) |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
|  |  | Введите путь файла с начальными данными  Результат решения первой задачи :  dva pyat chetire shest tri raz  Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Удачи |

Таблицы 3 – вторая задача

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Данные на выходе** |
| 1 | 2  0  Раз два три четыре пять шесть  Три че  1  D:\test.txt  нет | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла)  Введите предложение  Введите подстроку |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
|  |  | Результат решения второй задачи :  Строка "раз два три четыре пять шесть"  содержит подстроку "три че"  Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Удачи |
| 2 | 2  1  D:\second.txt  1  D:\test.txt  да | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла)  Введите путь файла с начальными данными  Результат решения второй задачи :  Строка "raz dva tri chetire pyat shest"  не содержит подстроку "razdva" |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
|  |  | Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово |

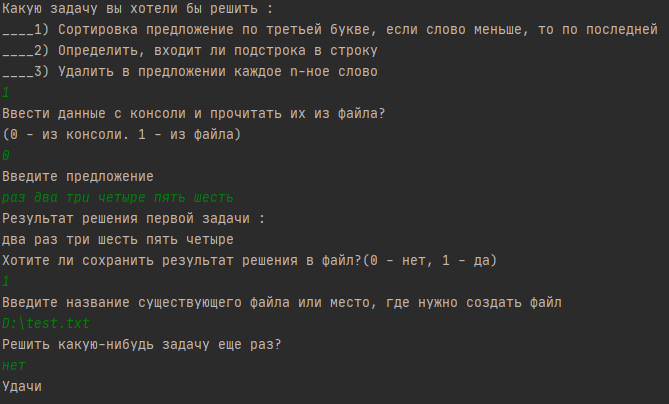
Таблица 4 - третья задача

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Данные на выходе** |
| 1 | 3  0  Раз два три четыре пять шесть семь  3  1  D:\test.txt  нет | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла)  Введите предложение |

Продолжение таблицы 4

|  |  |  |
| --- | --- | --- |
|  |  | Введите номер удаляемого слова  Результат решения третьей задачи :  раз два четыре пять семь  Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Удачи |
| 2 | 3  1  D:\third.txt  1  D:\test.txt  нет | Какую задачу вы хотели бы решить :  \_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней  \_\_\_\_2) Определить, входит ли подстрока в строку  \_\_\_\_3) Удалить в предложении каждое n-ное слово  Ввести данные с консоли и прочитать их из файла?  (0 - из консоли. 1 - из файла)  Введите путь файла с начальными данными  Результат решения третьей задачи :  raz tri pyat  Хотите ли сохранить результат решения в файл?(0 - нет, 1 - да)  Введите название существующего файла или место, где нужно создать файл  Решить какую-нибудь задачу еще раз?  Удачи |

6 Результаты тестирования



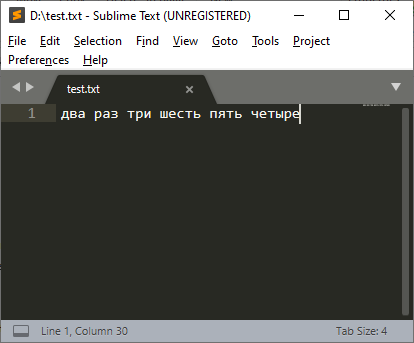


Рисунок 3 и 4 – Тест 1.1

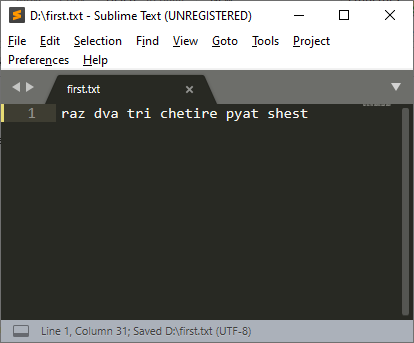
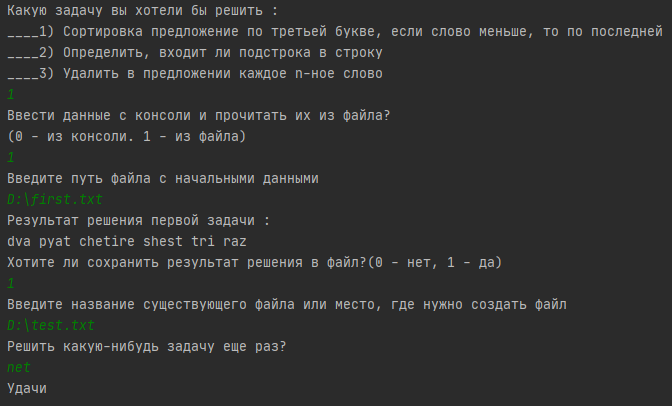
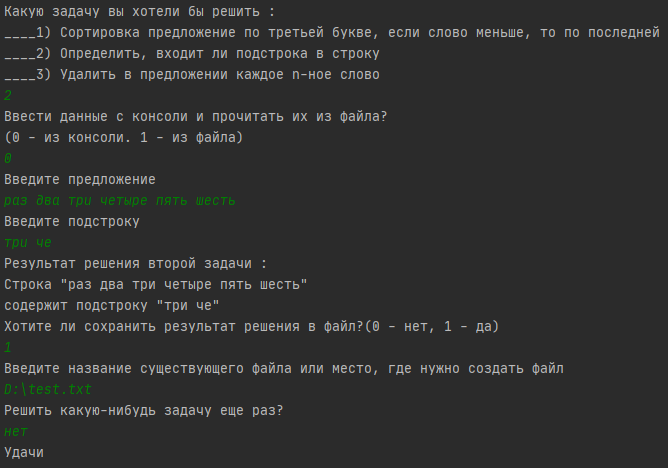


Рисунок 5 и 6 – Тест 1.2



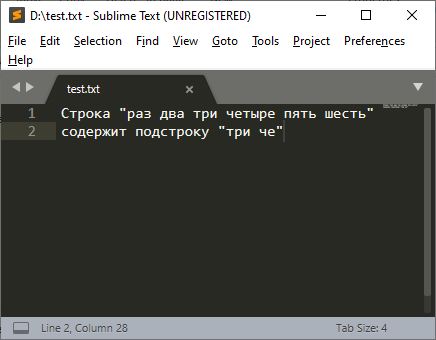


Рисунок 7 и 8 – Тест 2.1

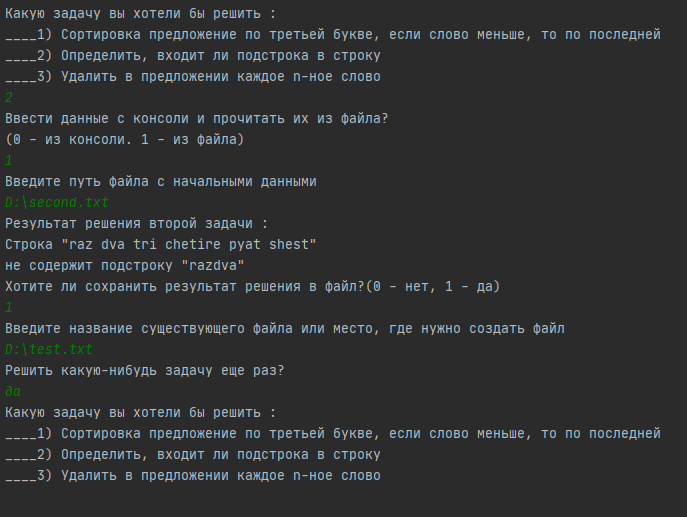
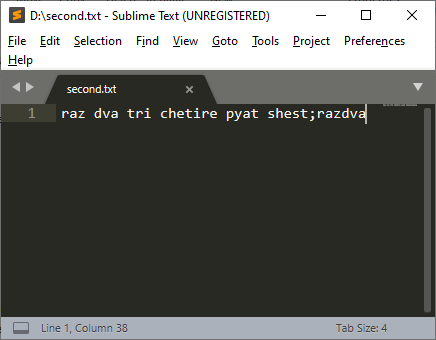


Рисунок 9 и 10 – Тест 2.2

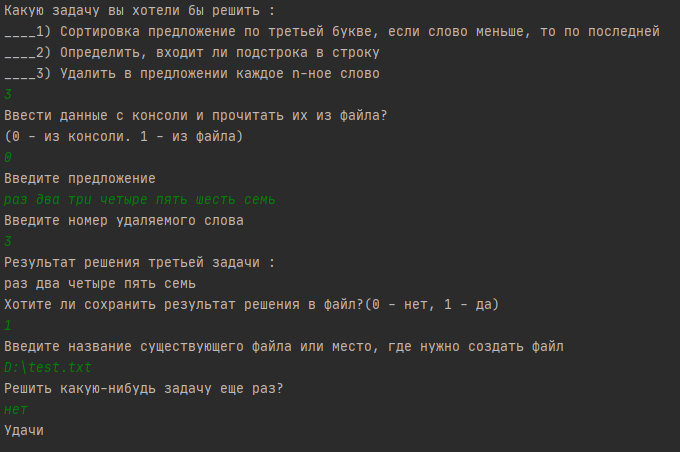
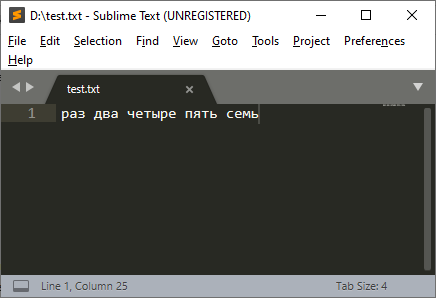
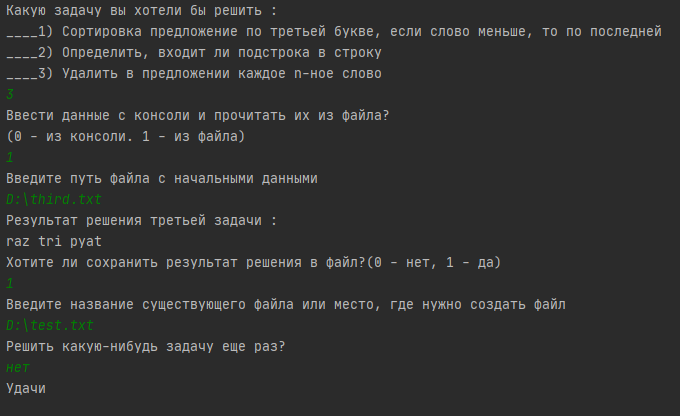
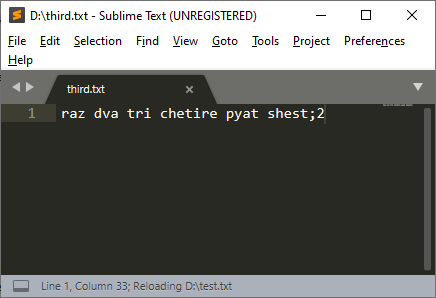


Рисунок 11 и 12 – Тест 3.1



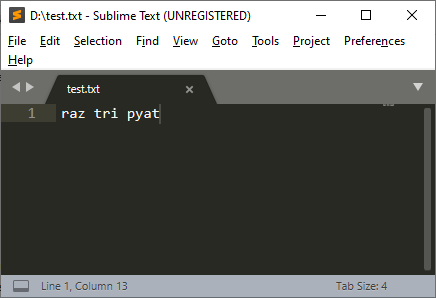


Рисунок 13,14 и 15 – Тест 3.2

7 Код программы

**Код класса Main**

package com.company;  
  
import java.util.Scanner;  
  
class Main{  
 static Scanner *in* = new Scanner(System.*in*);  
  
 public static void main(String[] args) {  
 String reply = "Yes";  
 while (reply.toLowerCase().equals("yes") || reply.toLowerCase().equals("y") ||  
 reply.toLowerCase().equals("да") || reply.toLowerCase().equals("1") ||  
 reply.toLowerCase().equals("+") || reply.toLowerCase().equals("lf") ||  
 reply.toLowerCase().equals("нуы") || reply.toLowerCase().equals("н") ||  
 reply.toLowerCase().equals("=")) { //проверка различных ответов пользователя на вопрос о повторном запуске программы  
  
 InteractMenu.*start*();  
 int number = InteractMenu.*taskNumber*();  
 Data data = InteractMenu.*getData*(number);  
 String res = InteractMenu.*solve*(data, number);  
 InteractMenu.*outData*(res); //вывод в файл, если надо пользователю  
  
 System.*out*.println("Решить какую-нибудь задачу еще раз?");  
 reply = *in*.nextLine(); //заново проийтись по программе или выйти из нее  
 }  
 InteractMenu.*wishLuck*();  
 }  
}

**Код класса Sentence**

package com.company;  
  
public class Sentence {  
  
 private final String[] stringArray;  
  
 public Sentence() { //базовый конструктор  
 this.stringArray = new String[]{"I", "am", "keen", "on", "programming", "on", "java"};  
 }  
  
 public Sentence(String[] stringArray) { //конструктор с входными данными  
 this.stringArray = stringArray;  
 }  
  
 public Sentence(Sentence sent){  
 this.stringArray = sent.stringArray;  
 }  
  
 public void swapValues(int ind1, int ind2) { //обмен переменных массива по индексам  
 String temp = stringArray[ind2];  
 stringArray[ind2] = stringArray[ind1];  
 stringArray[ind1] = temp;  
 }  
  
 public void sort() { //метод сортировки массива  
 // по третьей букве, если слово меньше трех букв, то по последней.  
 for (int i = 0; i < stringArray.length; i++)  
 {  
 for (int j = i; j < stringArray.length; j++)  
 {  
 String str1 = stringArray[i]; //слова из массива строк, чтобы  
 String str2 = stringArray[j]; //удобнее обращаться к их символам  
  
 if (str1.length() >= 3) //если 1 слово длиннее или равно 3 буквам  
 {  
 //если 2 слово длиннее или равно 3 буквам и 3-ья буква 1-го слово больше, чем 2-ая  
 if (str2.length() >= 3 && str1.charAt(2) > str2.charAt(2)) swapValues(i, j);  
 //иначе если 3-я буква 1-го слова больше последней буквы 2-го слова  
 else if (str2.length() < 3 && str1.charAt(2) > str2.charAt(str2.length() - 1)) swapValues(i ,j);  
 }  
 else  
 {  
 //аналогично, как выше  
 if (str2.length() >= 3 && str1.charAt(str1.length() - 1) > str2.charAt(2)) swapValues(i, j);  
 else if (str2.length() < 3 && str1.charAt(str1.length() - 1) > str2.charAt(str2.length() - 1)) swapValues(i ,j);  
 }  
 }  
 }  
 }  
  
 public boolean ifContains(String subStr) //метод на проверку, содержит ли строка подстроку  
 {  
 int count = 0; //счетчик для того, чтобы считать текущий индекс подстроки  
 String sent = new Sentence(stringArray).toString(); //получение строки из объекта, потому что здесь нужна именно строка, а не массив строк  
 for (int i = 0; i < sent.length(); i++)  
 {  
 if (sent.charAt(i) == subStr.charAt(count)) //если текущий символ строки совпадает с символом подстроки  
 {  
 count++;  
 //если подстрока полностью содержится в строке, т.е. количество совпадающих символов в строке и подстроке равно длине подстроки  
 if (count == subStr.length())  
 {  
 return true;  
 }  
 }  
 else count = 0; //иначе подстрока не содержится полностью и индекс подстроки обнуляется  
 }  
 return false;  
 }  
  
 public void delWords(int number) //удаление каждого number-го слова в массиве  
 {  
 if (number < 1 || number > stringArray.length) {  
 System.*out*.println("В предложении нет слова с таким номером");  
 return;  
 }  
 int count = 1;  
 for (int i = 0; i < stringArray.length; i++){  
 if (count++ % number == 0) stringArray[i] = "";  
 }  
 }  
  
 @Override //переопределение метода toString() для текущего класса  
 public String toString() {  
 StringBuilder out = new StringBuilder();  
 for (String s : stringArray) {  
 out.append(s);  
 if (!s.equals("")) out.append(" ");  
 }  
 out.replace(out.length() - 1, out.length(), "");  
 return out.toString();  
 }  
  
 public String getStringOfArray() {  
 Sentence sent = new Sentence(this.stringArray);  
 return sent.toString();  
 } //геттер для поля класса  
}

**Код класса InteractMenu**

package com.company;  
  
import java.util.InputMismatchException;  
import java.util.Scanner;  
  
public class InteractMenu {  
  
 public static void start() {  
 System.out.println("Какую задачу вы хотели бы решить : ");  
 System.out.println("\_\_\_\_1) Сортировка предложение по третьей букве, если слово меньше, то по последней");  
 System.out.println("\_\_\_\_2) Определить, входит ли подстрока в строку");  
 System.out.println("\_\_\_\_3) Удалить в предложении каждое n-ное слово");  
 }  
  
 public static int taskNumber() {  
 String num = InteractMenu.ifNumber(); //ввод и проверка на число  
 while (num == null || !(num.equals("1") || num.equals("2") || num.equals("3"))) { //если введено было не число или неверный номер задания  
 System.out.println("Вы ввели не число или нет задания с таким номером, повторите ввод");  
 num = InteractMenu.ifNumber();  
 }  
 return Integer.parseInt(num); //номер задания  
 }  
  
 public static String ifNumber() { //проверяет, число ли введено вообще  
 Scanner in = new Scanner(System.in);  
 int number;  
 try {  
 number = in.nextInt(); //пробует ввести число  
 return String.valueOf(number); //String, чтобы можно было возвратить null  
 }  
 catch (InputMismatchException e) { //если не получилось считать число, то возвратится null  
 return null;  
 }  
 }  
  
 public static String readWay() { //определяет, каким путем получать входные данные  
 Scanner in = new Scanner(System.in);  
 System.out.println("Ввести данные с консоли и прочитать их из файла?\n" +  
 "(0 - из консоли. 1 - из файла)");  
 String reply = in.nextLine();  
 while (!reply.equals("0") && !reply.equals("1")) { //проверяет, введено ли 0 или 1  
 System.out.println("Введите 0 или 1");  
 reply = in.nextLine(); //если введены не 0 и 1, то запрашивает проверку снова  
 }  
 if (reply.equals("0")) return "console";  
 else return "file";  
 }  
  
 public static boolean ifWrite() {  
 Scanner in = new Scanner(System.in);  
 System.out.println("Хотите ли сохранить результат решения в файл?" +  
 "(0 - нет, 1 - да)");  
 String reply = in.nextLine();  
 while (!reply.equals("0") && !reply.equals("1")) { //пока ответ пользователя не будет равен 0 или 1  
 System.out.println("Введите 0 или 1");  
 reply = in.nextLine();  
 }  
 return !reply.equals("0"); //если пользователь не хочет делать запись данных в файл  
 }  
  
 public static void outData(String data) {  
 if (ifWrite()) {  
 FileHandler.writeFile(data);  
 }  
 }  
  
 public static Data getData(int number) {  
 String way = InteractMenu.readWay();  
 Data data;  
 if (way.equals("console")) { //если считывать с консоли  
 data = InteractMenu.inputData(number);  
 while (data == null) { //если входные данные ведены неверно  
 System.out.println("Повторите ввод заново");  
 data = InteractMenu.inputData(number);  
 }  
 }  
 else { //иначе если читать с файла  
 String fileData = FileHandler.readFile();  
 data = FileHandler.fileInformationSplit(fileData, number);  
 }  
 return data;  
 }  
  
 public static Data inputData(int num) { //ввод данных с консоли для задачи  
 Scanner in = new Scanner(System.in);  
 String input;  
 Data data;  
 switch (num) {  
 case 1 -> { //если первая задача  
 System.out.println("Введите предложение");  
 input = in.nextLine();  
 data = new Data(input);  
 return data;  
 }  
 case 2 -> { //если вторая задача  
 System.out.println("Введите предложение");  
 input = in.nextLine();  
 System.out.println("Введите подстроку");  
 String subString = in.nextLine();  
 data = new Data(input, subString);  
 return data;  
 }  
 case 3 -> { //если третья задача  
 System.out.println("Введите предложение");  
 input = in.nextLine();  
 System.out.println("Введите номер удаляемого слова");  
 String temp;  
 int number;  
 temp = ifNumber(); //сразу проверка на то, число ли введено  
 if (temp == null) break; //если нет, то выходим из кейса  
 number = Integer.parseInt(temp); //преобразование строки в число, так как уже понятно, что это число  
 if (number < 1 || number > input.split(" ").length) { //проверка на то, есть ли слово с таким номером в предложении  
 System.out.println("В предложении нет слова с таким номером");  
 break;  
 }  
 data = new Data(input, number);  
 return data;  
 }  
 default -> { //если введен неверный номер задачи  
 System.out.println("Вы ввели неверный номер задачи");  
 System.*out*.println("Попробуйте еще раз");  
 }  
 }  
 return null; //если введен неверный номер задачи, то просто возврат null  
 }  
  
 public static String solve(Data data, int number) {  
 String res = ""; //данные для подачи на запись в файл  
 if (data != null) {  
 String[] sentence; //переменные  
 String subString; //для  
 int count; //задач  
 sentence = data.getSentence().split(" "); //первая строка для всех одинакова, поэтому сразу ее назначаем  
  
 if (number == 1) { //если задача 1-ая, 2-ая или 3-я  
 res = InteractMenu.solveFirst(sentence).getStringOfArray();  
 } else if (number == 2) {  
 subString = data.getSubString();  
 res = InteractMenu.solveSecond(sentence, subString);  
 } else {  
 count = data.getNumber();  
 res = InteractMenu.solveThird(sentence, count).getStringOfArray();  
 }  
 }  
 return res;  
 }  
  
 public static Sentence solveFirst(String[] sentence) { //решение первой задачи  
 Sentence sent = new Sentence(sentence);  
 sent.sort();  
 System.out.println("Результат решения первой задачи :");  
 System.out.println(sent);  
 return sent; //возврат данных здесь и в последующих 2-ух задачах для возможной записи их в файл  
 }  
  
 public static String solveSecond(String[] sentence, String subString) { //решение второй задачи  
 Sentence sent = new Sentence(sentence);  
 System.out.println("Результат решения второй задачи :");  
 String ans;  
 if (sent.ifContains(subString)) ans = "Строка \"" + sent + "\"" +  
 "\nсодержит подстроку \"" + subString + "\"";  
 else ans = "Строка \"" + sent + "\"" +  
 "\nне содержит подстроку \"" + subString + "\"";  
 System.out.println(ans);  
 return ans;  
 }  
  
 public static Sentence solveThird(String[] sentence, int number) { //решение третьей задачи  
 Sentence sent = new Sentence(sentence);  
 sent.delWords(number);  
 System.out.println("Результат решения третьей задачи :");  
 System.out.println(sent);  
 return sent;  
 }  
  
 public static void wishLuck() {  
 System.out.println("Удачи");  
 }  
}

**Код класса FileHandler**

package com.company;  
  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.util.Scanner;  
  
public class FileHandler {  
 public static String readFile() { //выполняет чтение из файла и возвращает данные из него  
 Scanner in = new Scanner(System.in);  
 boolean flag = true;  
 StringBuilder res = new StringBuilder(); //производится много конкатенаций строк, поэтому лучше использовать StringBuilder  
 while (flag) { //пока небудет введен верный путь файла  
 try {  
 System.out.println("Введите путь файла с начальными данными");  
 String path = in.nextLine();  
 FileInputStream fis = new FileInputStream(path); //поток чтения из файла  
 flag = false;  
 int symbol;  
 while ((symbol = fis.read()) != -1) { //побайтное чтение из файла  
 res.append((char) symbol);  
 }  
 fis.close(); //закрытие файла  
 if (res.length() == 0) return null; //если файл пустой, возвращается null  
 } catch (IOException e) {  
 System.out.println("Файла по такому пути не существует, введите путь заново");  
 }  
 }  
 return res.toString(); //возврат StringBuilder в виде String  
 }  
  
 public static Data fileInformationSplit(String fileData, int taskNumber) { //разделение информации из файла (она изначально должна быть разделена знаками ";")  
 if (fileData == null) { //если файл пустой, то даже нет смысла разделить ничего  
 System.out.println("Файл пустой");  
 return null;  
 }  
 String[] arrData = fileData.split(";"); //разделение строк по символу их разделения  
 Data data; //объект моего класса данных  
 if (arrData.length == 1 && taskNumber == 1) { //если введена только одна строка без единых знаков ";" (т.е. для первого задания)  
 data = new Data(arrData[0]); //создание экземпляра класса данных, здесь и далее (см. Data.java)  
 }  
 else if (arrData.length == 2) { //для 2-го и 3-го задания, т.к. там две строки в обоих заданиях  
 String temp = arrData[1]; //второе предложение  
 int number;  
 if (taskNumber == 3) {  
 try {  
 number = Integer.parseInt(temp); //попытка преобразовать 2-ое предложение в число  
 data = new Data(arrData[0], number); //запись данных для задания  
 } catch (NumberFormatException e) {  
 System.out.println("В файле введены неверные входные данные для третьего задания");  
 return null; //иначе нельзя решить данное задание с такими входными данными  
 }  
 }  
 else {  
 data = new Data(arrData[0], temp); //иначе просто добавить это как строку  
 }  
 }  
 else {  
 System.*out*.println("Файл должен содержать строго только входные данные для решения задач, разделенные знаком \";\"" +  
 "или для данной задачи введеные некорректные данные в файле");  
 data = null; //нельзя решить задачу с такими входными данными  
 }  
 return data; //возврат данных для решения задачи  
 }  
  
 public static void writeFile(String data) { //запись данных в файл  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("Введите название существующего файла или место, где нужно создать файл");  
 String path = in.nextLine(); //путь файла  
 try {  
 if (path.equals("0")) return; //если пользователь ввел 0 в catch  
 FileOutputStream fos = new FileOutputStream(path); //поток вывода данных в файл  
 PrintWriter pw = new PrintWriter(fos); //запись текстовых данных в файл, чтобы кодировка была русская,  
 //иначе будет происходить побайтовая запись данных и будут выводится непонятные символы  
 pw.write(data); //запись данных в файл  
 pw.close();  
 fos.close(); //закрытие потоков вывода  
 }catch (IOException e) {  
 System.out.println("Вы ввели название каталога или файл не может быть создан или открыт по некоторым причинам" +  
 "повторите ввод");  
 path = in.nextLine(); //иначе просьба выполнить повторый ввод и возможность отменить это действие  
 }  
 }  
}

**Код класса Data**

package com.company;  
  
public class Data {  
 String sentence;  
 String subString;  
 int number;  
  
 public Data(String sentence) {  
 this.sentence = sentence;  
 }  
  
 public Data(String sentence, String subString) {  
 this.sentence = sentence;  
 this.subString = subString;  
 }  
  
 public Data(String sentence, int number) {  
 this.sentence = sentence;  
 this.number = number;  
 }  
  
 public String getSentence() {  
 return this.sentence;  
 }  
  
 public String getSubString() {  
 return this.subString;  
 }  
  
 public int getNumber() {  
 return this.number;  
 }  
}

Список использованных источников

1. Объектно-ориентированное программирование. Метод. указания по

выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск : Изд-во

ИРНИТУ, 2017. – 24 c.