Министерство науки и образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

Высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

РАЗРАБОТКА КОНСОЛЬНОГО ПРИЛОЖЕНИЯ

Отсчёт по лабораторной работе №1

Вариант 14

По дисциплине: «Объектно-ориентированное программирование»

Выполнил студент

группы АСУб-19-1 \_\_\_\_\_\_\_\_\_\_\_ Д. А. Вологдин

Принял

Доцент \_\_\_\_\_\_\_\_\_\_\_  Т.В. Маланова

г. Иркутск 2020

**Оглавление**

[1 Постановка задачи 3](#_Toc32754747)

[2 Блок-схема алгоритма 4](#_Toc32754748)

[3 Таблица спецификации 7](#_Toc32754749)

[4 Таблица тестов 8](#_Toc32754750)

[5 Результаты тестирования 9](#_Toc32754751)

[6 Исходный код 10](#_Toc32754752)

[Список использованных источников 11](#_Toc32754753)

1 Постановка задачи

Разработать консольное приложение на языке java, реализующее решение задачи в соответствии с индивидуальным заданием. Исходные данные для задачи вводятся как параметры при запуске программы из командной строки, либо – как значения, инициализирующие переменные. Вывод результатов осуществляется в консоль.

1. Отсортируйте слова в массиве по их длине.
2. Подсчитайте количество предложений в заданном тексте.
3. Дана строка, содержащая слова, разделённые пробелами, запятыми, восклицательными знаками, вопросительными знаками. Отсортируйте слова по их длине.

Математическая модель:

1. Разделить строку на отдельные слова, составить из полученных слов массив и отсортировать полученный массив по длине слов
2. С помощью цикла проверить каждый символ текста, если будет найден символ заканчивающий предложение, прибавить к количеству предложений 1.
3. С помощью цикла проверить строку, и при нахождении разделяющего символа убрать этот символ путём конкатенации части строки до символа и части строки после символа. Разделить строку на слова, составить из слов массив и отсортировать его по длине слов.

2 Блок-схема алгоритма

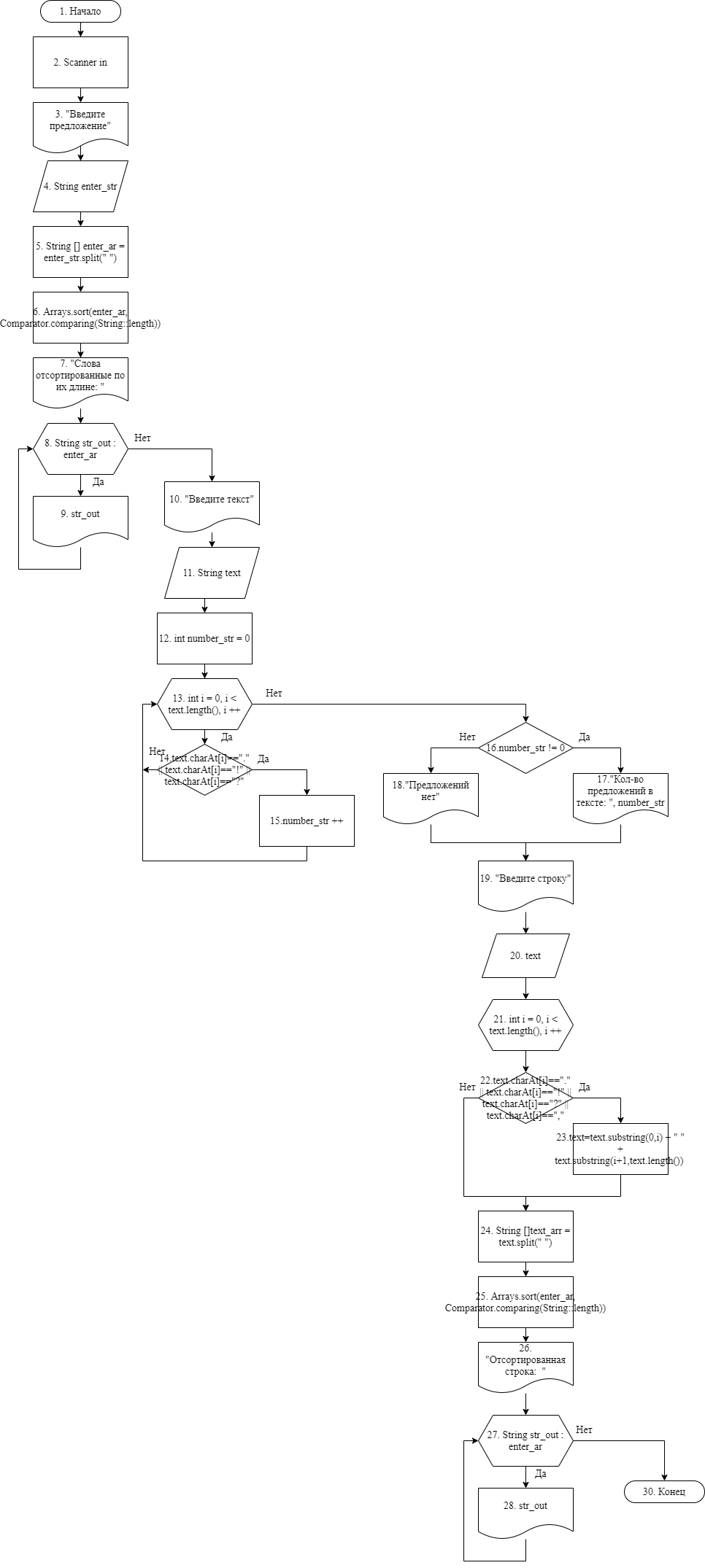


Рисунок 1 – Блок-схема

3 Таблица спецификации

Таблица 1 – Таблица спецификаций

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Описание переменных | Тип данных | ОДЗ |
| 1 | Enter\_str | Исходная строка | String | [‘A’..’z’] |
| 2 | Enter\_ar | Массив слов исходной строки | String[] | [‘A’..’z’] |
| 3 | text | Исходный текст, или строка | String | [‘A’..’z’] |
| 4 | Text\_arr | Массив слов исходной строки или текста | String | [‘A’..’z’] |
| 5 | Str\_out | Временная строка для вывода массива строк | String | [‘A’..’z’] |
| 6 | Number\_str | Количество предложений в текстк | int | [-2147483647.. -2147483647] |
| 7 | i | Индекс символа в строке | int | [1..str.length()] |

4 Таблица тестов

Таблица 2 – Таблица тестов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание | Входные данные | | Выходные данные |
| 1 | Сортировка слов в массиве  (a\_task) | Sentence | «Я слов на ветер не бросаю. Это мой путь ниндзя» | «Слова отсортированные по их длине: Я на не Это мой слов путь ветер ниндзя бросаю.» |
| 2 | Подсчёт кол-ва предложений в тексте  (b\_task) | Sentence | «Ничего не понимаю! Ладно, притворюсь, что понял.» | Кол-во предложений в тексте: 2 |
| 3 | Сортировка слов в строке  (c\_task) | Sentence | “На!ошибках.учатся,после?ошибок.лечатся.” | «Отсортированная строка: На после учатся ошибок ошибках лечатся» |
| 4 | Сортировка слов в массиве  (a\_task) | Sentence | «Только начнешь работать, обязательно кто-нибудь разбудит.» | «Слова отсортированные по их длине: Только начнешь работать, разбудит. кто-нибудь обязательно» |
| 5 | Подсчёт кол-ва предложений в тексте  (b\_task) | Sentence | «Мало сказать глупость, надо чтобы в нее поверили. Боль — это боль, как ее не назови. Разве она становится слабее от того, что ты думаешь, что она нереальна?» | Кол-во предложений в тексте: 3 |
| 6 | Сортировка слов в строке  (c\_task) | Sentence | «Люди?становятся?сильнее,благодаря?воспоминаниям,которые,не,могут,забыть.Это!и!есть.взросление.» | «Отсортированная строка: и не Это Люди есть могут забыть сильнее которые благодаря становятся взросление воспоминаниям» |

5 Результаты тестирования

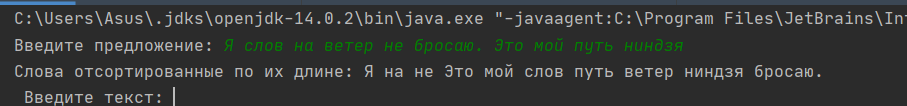


Рисунок 2.1 – Тест 1

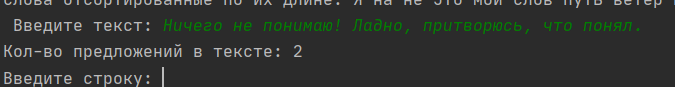


Рисунок 2.2 – Тест 1

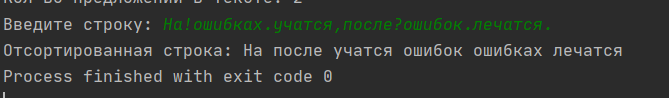


Рисунок 2.3 – Тест 1

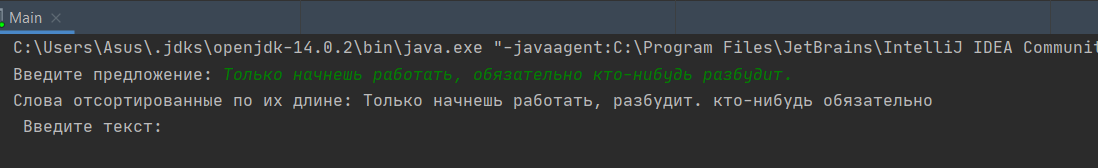


Рисунок2.4 – Тест 2

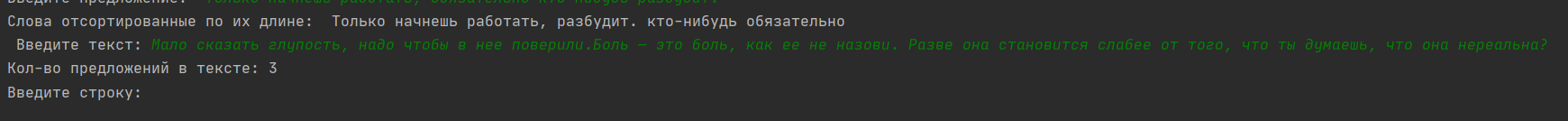


Рисунок 2.5 – Тест 2

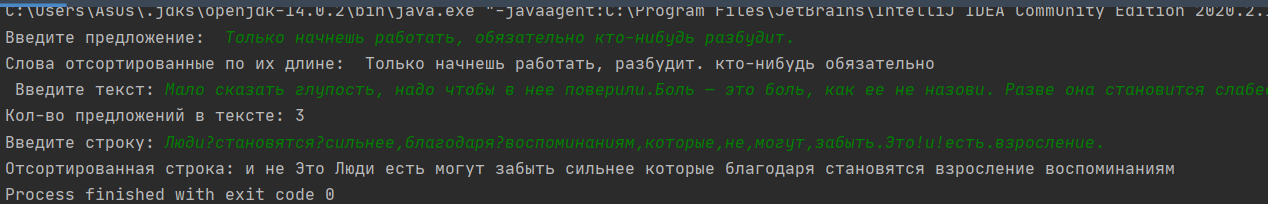


Рисунок 2.6 – Тест 2

6 Исходный код

package L114;  
import java.util.Arrays;  
import java.util.Comparator;  
import java.util.Scanner;  
  
public class L114 {  
  
 public static void main(String[] args){  
 //Создаём объект класса Scanner для считывания строк  
 Scanner in = new Scanner(System.in);  
 System.out.print("Введите предложение: ");  
 String enter\_str = in.nextLine();  
 //Преобразуем строку в массив  
 String [] enter\_ar = enter\_str.split(" ");  
 //Сортируем массив по длине слов  
 Arrays.sort(enter\_ar, Comparator.comparing(String::length));  
 System.out.print("Слова отсортированные по их длине: ");  
 //Выводим полученный массив  
 for (String str\_out : enter\_ar) {  
 System.out.print(str\_out + " ");  
 }  
 System.out.print("\n Введите текст: ");  
 String text = in.nextLine();  
 int number\_str = 0;  
 //Проверяем каждый элемент строки  
 for (int i = 0; i < text.length(); i++){  
 if (text.charAt(i) == '.' || text.charAt(i) == '!' || text.charAt(i) == '?'){  
 number\_str ++;  
 }  
 }  
 if (number\_str != 0) System.out.printf("Кол-во предложений в тексте: %d \n", number\_str);  
 else System.out.print("Предложений нет \n");  
 System.out.print("Введите строку: ");  
 text = in.nextLine();  
 //Поиск ненужных симовлов в строке и их удоление  
 for (int i = 0; i < text.length(); i++){  
 if (text.charAt(i) == '.' || text.charAt(i) == '!' || text.charAt(i) == '?' || text.charAt(i) == ','){  
 text = text.substring(0, i) + " " +text.substring(i + 1, text.length());  
 }  
 }  
 //Из готовой строки делаем массив и сортируем слова в нём  
 String [] text\_arr = text.split(" ");  
 Arrays.sort(text\_arr, Comparator.comparing(String::length));  
 System.out.print("Отсортированная строка: ");  
 //Вывод отсортированной строки  
 for (String str\_out : text\_arr) {  
 System.out.print(str\_out + " ");  
 }  
 in.close();  
 }  
}

Список использованных источников

1. Объектно-ориентированное программирование. Метод. указания по

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

ИРНИТУ, 2017. – 24 c.