|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2** | |
| **по дисциплине** | |
| **«**Объектно-ориентированное программирование**»**  **Тема: «**Строки в Java. Применение регулярных выражений для поиска заданных слов**»** | |
|  | |
| Выполнил студент группы ИКБО-17-18 | Орлова Е.М. |
| Принял преподаватель | Баранова И.А. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *(подпись руководителя)* |

Условие задачи

Разработать программу, которая обрабатывает строки в Java.

Постановка задачи

Дано:

Для типа String:

Дан художественный текст. По количеству используемых в тексте слов можно с некоторой вероятностью определить автора. Равными словами считать слова, содержащие строчные и прописные буквы (например, сЛоВо и СЛОВО и слово).

* Преобразовать все слова к одному формату: либо строчные буквы, либо прописные.
* Сформировать массив из слов.
* Сформировать массив вхождений каждого найденного слова в текст.
* Определить, сколько слов начинаются с заданного префикса.

Для типа StringBuilder:

Дан текст, состоящий из предложений, заканчивающиеся символами конца предложений. Предложение состоит из слов. Словами могут быть числа.

* Найти в тексте двузначные числа и заменить их строкой из символов + длиной равной двузначному числу.
* Добавить в самое короткое предложение текста одно слово КОНЕЦ.
* Найти трехзначные числа и заменить из обратными числами, т.е.число 123 заменить на 321.

Для регулярных выражений.

* Определите, что строка является корректным email-ом.
* Дан текст. Сформировать список всех электронных адресов, содержащихся в этом тексте

Разработка тестов

Для типа String задание №1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | Текст на проверку | ТЕКСТ НА ПРОВЕРКУ | ТЕКСТ НА ПРОВЕРКУ | Пройден |
| 2. | Шла Саша по шоссе | ШЛА САША ПО ШОССЕ | ШЛА САША ПО ШОССЕ | Пройден |
| 3. | И сосала соску | И СОСАЛА СУШКУ | И СОСАЛА СУШКУ | Пройден |
| 4. | Привет мир | ПРИВЕТ МИР | ПРИВЕТ МИР | Пройден |
| 5. | Я люблю java | Я ЛЮБЛЮ JAVA | Я ЛЮБЛЮ JAVA | Пройден |

Для типа String задание №2:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | Текст на проверку | [Текст, на, проверку] | [Текст, на, проверку] | Пройден |
| 2. | Шла Саша по шоссе | [Шла, Саша, по, шоссе] | [Шла, Саша, по, шоссе] | Пройден |
| 3. | И сосала соску | [И, сосала, сушку] | [И, сосала, сушку] | Пройден |
| 4. | Привет мир | [Привет, мир] | [Привет, мир] | Пройден |
| 5. | Я люблю java | [Я, люблю, java] | [Я, люблю, java] | Пройден |

Для типа String задание №3:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | Текст на проверку | [0, 6, 9] | [0, 6, 9] | Пройден |
| 2. | Шла Саша по шоссе | [0, 4, 9, 12] | [0, 4, 9, 12] | Пройден |
| 3. | И сосала соску | [0, 2, 9] | [0, 2, 9] | Пройден |
| 4. | Привет мир | [0, 7] | [0, 7] | Пройден |
| 5. | Я люблю java | [0, 2, 8] | [0, 2, 8] | Пройден |

Для типа String задание №4:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | | | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | Слово1 слово2 слово3 слово4 | Сло | 4 | | 4 | Пройден |
| 2. | Привет мир привет | При | 2 | | 2 | Пройден |
| 3. | Мама мыла раму | М | 2 | | 2 | Пройден |
| 4. | Текст на тест | Те | 2 | | 2 | Пройден |
| 5. | Река дон большая река | рек | 2 | | 2 | Пройден |

Для типа StringBuilder задание №1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | 123 15 256 | 123 +++++++++++++++ 256 | 123 +++++++++++++++ 256 | Пройден |
| 2. | 3 6 10 | 3 6 ++++++++++ | 3 6 ++++++++++ | Пройден |
| 3. | 75396 слово 10 956 | 75396 слово ++++++++++ 956 | 75396 слово ++++++++++ 956 | Пройден |
| 4. | Текст без чисел | Текст без чисел | Текст без чисел | Пройден |
| 5. | 10 | ++++++++++ | ++++++++++ | Пройден |

Для типа StringBuilder задание №2:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | Привет. Мир. | Привет. Мир КОНЕЦ. | Привет. Мир КОНЕЦ. | Пройден |
| 2. | Текст. на проверку. | Текст КОНЕЦ. на проверку. | Текст КОНЕЦ. на проверку. | Пройден |
| 3. | Шла. Саша по. шоссе. | Шла КОНЕЦ. Саша по. шоссе. | Шла КОНЕЦ. Саша по. шоссе. | Пройден |
| 4. | И сосала. сушку. | И сосала. сушку КОНЕЦ. | И сосала. сушку КОНЕЦ. | Пройден |
| 5. | Я люблю java. | Я люблю java КОНЕЦ. | Я люблю java КОНЕЦ. | Пройден |

Для типа StringBuilder задание №3:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | 256 852 13 | 652 258 13 | 652 258 13 | Пройден |
| 2. | Текст без чисел | Текст без чисел | Текст без чисел | Пройден |
| 3. | 753 15822 951 | 357 15228 159 | 357 15228 159 | Пройден |
| 4. | 100 641 слово | 001 146 слово | 001 146 слово | Пройден |
| 5. | Текст 188 | Текст 881 | Текст 881 | Пройден |

Для типа регулярных выражений:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Номер теста* | *Исходные данные* | *Эталон результата* | *Результат программы* | *Отметка о правильном прохождении теста* |
| 1. | [name@email.ru](mailto:name@email.ru) все | Список Email'ов:  name@email.ru | Список Email'ов:  name@email.ru | Пройден |
| 2. | [Name1@email.ru](mailto:Name1@email.ru) и name2@email.ru | Список Email'ов:  Name1@email.ru  name2@email.ru | Список Email'ов:  Name1@email.ru  name2@email.ru | Пройден |
| 3. | Без имейла | Список Email'ов: | Список Email'ов: | Пройден |
| 4. | [text@text.com](mailto:text@text.com) и text@text.ru | Список Email'ов:  text@text.com  text@text.ru | Список Email'ов:  text@text.com  text@text.ru | Пройден |
| 5. | [rand@mail.com](mailto:rand@mail.com) конец | Список Email'ов:  rand@mail.com | Список Email'ов:  rand@mail.com | Пройден |

Исходный код программы

Class Main:

package com.company;

import java.lang.reflect.Array;

import java.util.Arrays;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

int test = 8;

Scanner in = new Scanner(System.in);

while (test != 0) {

try {

System.out.print("[0] - Завершение работы" + '\n' +

"[1] - Вариант на String" + '\n' +

"[2] - Ванриант на StringBuffer" + '\n' +

"[3] - Вариант на регулярные выражения" + '\n' +

"Выберите вариант: ");

test = in.nextInt();

switch (test) {

case 0:

System.out.print("End of working");

break;

case 1:

testStringClass(in);

break;

case 2:

testStringBuilderClass(in);

break;

case 3:

testRegularExpressionsClass(in);

break;

default:

System.out.println("Вы ввели недопустимую конструкцию!");

break;

}

} catch (Exception e) {

System.out.println(e);

}

}

}

static void testStringClass(Scanner in) throws Exception {

StringWork object = new StringWork();

int task = 8;

String str;

while (task != 0){

System.out.print("[0] - Предыдущее меню" + '\n' +

"[1] - Ввод строки" + '\n' +

"[2] - Вывод строки" + '\n' +

"[3] - Преобразовать все слова к одному формату: либо строчные буквы,\n" +

"либо прописные" + '\n' +

"[4] - Сформировать массив из слов" + '\n' +

"[5] - Сформировать массив вхождений каждого найденного слова в текст" + '\n' +

"[6] - Определить, сколько слов начинаются с заданного префикса" + '\n' +

"Выберете задание: ");

task = in.nextInt();

switch(task){

case 0:

System.out.println("Возвращение к предыдущему меню...");

break;

case 1:

System.out.println("Введите текст: ");

in.nextLine();

str = in.nextLine();

object.setText(str);

break;

case 2:

System.out.println(object.getText());

break;

case 3:

object.allTo(in);

System.out.println("Итоговая строка:\n" +object.getText());

break;

case 4:

String sArr[] = object.getArr();

System.out.println("Массив строк:\n" + Arrays.toString(sArr));

break;

case 5:

int iArr[] = object.getArrInd();

System.out.println("Массив индексов вхождений слов:\n" + Arrays.toString(iArr));

break;

case 6:

System.out.println("Введите префикс:");

in.nextLine();

str = in.nextLine();

System.out.println(object.wordCount(str).toString() + " слов начинается с префикса \"" + str + "\"");

break;

default:

System.out.println("Вы ввели недопустимую конструкцию!");

break;

}

}

}

static void testStringBuilderClass(Scanner in) {

StringBuilderWork object = new StringBuilderWork();

int task = 8;

while (task != 0){

System.out.print("[0] - Предыдущее меню" + '\n' +

"[1] - Ввод строки" + '\n' +

"[2] - Вывод строки" + '\n' +

"[3] - Найти в тексте двузначные числа и заменить их строкой из\n" +

"символов + длиной равной двузначному числу" + '\n' +

"[4] - Добавить в самое короткое предложение текста одно слово\n" +

"КОНЕЦ" + '\n' +

"[5] - Найти трехзначные числа и заменить из обратными числами,\n" +

"т.е.число 123 заменить на 321" + '\n' +

"Выберете задание: ");

task = in.nextInt();

switch(task){

case 0:

System.out.println("Возвращение к предыдущему меню...");

break;

case 1:

System.out.println("Введите текст: ");

in.nextLine();

String str = in.nextLine();

object.setText(new StringBuilder(str));

break;

case 2:

System.out.println(object.getText());

break;

case 3:

object.setPlus();

System.out.println("Строка с замененными числами:\n" + object.getText());

break;

case 4:

object.addKONEC();

System.out.println("Строка с добавленым концом:\n" + object.getText());

break;

case 5:

object.changeNum();

System.out.println("Строка с изсменнеными числами:\n" + object.getText());

break;

default:

System.out.println("Вы ввели недопустимую конструкцию!");

break;

}

}

}

static void testRegularExpressionsClass(Scanner in) {

RegularExpression object = new RegularExpression();

int task = 8;

while (task != 0){

System.out.print("[0] - Предыдущее меню" + '\n' +

"[1] - Ввод строки" + '\n' +

"[2] - Вывод строки" + '\n' +

"[3] - Дан текст. Сформировать список всех электронных адресов,\n" +

"содержащихся в этом тексте." + '\n' +

"Выберете задание: ");

task = in.nextInt();

switch (task){

case 0:

System.out.println("Возвращение к предыдущему меню...");

break;

case 1:

System.out.println("Введите текст: ");

in.nextLine();

String str = in.nextLine();

object.setText(str);

break;

case 2:

System.out.println("Ваш текст:\n" + object.getText());

break;

case 3:

System.out.println("Список Email\'ов:\n" + object.findEmail());

break;

default:

System.out.println("Вы ввели недопустимую конструкцию!");

break;

}

}

}

}

**Class RegularExpression**:

**package** com.company;  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
**public class** RegularExpression {  
 **private** String **text**;  
 **private** Pattern **pattern** = Pattern.*compile*(**"\\w+@\\w+\\.(com|ru)"**);  
  
 **public void** setText(String str){ **text** = str; }  
 **public** String getText(){**return text**;}  
  
 **public** String findEmail(){  
 Matcher m = **pattern**.matcher(**text**);  
 String str = **""**;  
 **while** (m.find()){  
 str+= **text**.substring(m.start(), m.end()) + **"\n"**;  
 }  
 *//str += m.toString() + "\n";  
 //text email@gmail.com email@mail.ru some text some email@il.ru* **return** str;  
 }  
}

Class StringWork:

**package** com.company;  
  
**import** java.util.Scanner;  
  
**public class** StringWork {  
 **private** String **text** = **""**;  
 **private int register**;*//Нужна для корректной работы функции поиска слов с префиксами, потому что у пользователя есть выбор* **public void** setText(String str){**text** = str;}  
 **public** String getText() {**return text**;}  
  
 **public void** allTo(Scanner in){  
 System.***out***.print(**"[0] - все строчные\n"** +  
 **"[1] - Все Прописные\n"** +  
 **"Выберите вариант: "**);  
 **register** = in.nextInt();  
 **switch** (**register**){  
 **case** 0:  
 **text** = **text**.toLowerCase();  
 **break**;  
 **case** 1:  
 **text** = **text**.toUpperCase();  
 **break**;  
 }  
 }  
 **public** String []getArr(){  
 String buf[] = **text**.split(**" "**);  
 **return** buf;  
 }  
 **public int**[] getArrInd(){  
 String buf[] = getArr();  
 **int** arr[] = **new int**[buf.**length**];  
 arr[0] = **text**.indexOf(buf[0]);  
 **for** (**int** i = 1; i < buf.**length**; i++){  
 arr[i] = **text**.indexOf(buf[i], arr[i-1] + 1);  
 }  
 **return** arr;  
 }  
 **public** Integer wordCount(String prefix){  
 **int** count = 0;  
 **switch** (**register**){  
 **case** 0:  
 prefix = prefix.toLowerCase();  
 **break**;  
 **case** 1:  
 prefix = prefix.toUpperCase();  
 **break**;  
 }  
 **for** (**int** i = 0; i < **text**.length(); i++){  
 i = **text**.indexOf(prefix, i);  
 **if** (i > -1)  
 count++;  
 **else  
 return** count;  
 }  
 **return** count;  
}  
}

Class StringBilderWork:

**package** com.company;  
  
**public class** StringBuilderWork {  
 **private** StringBuilder **text**;  
 **public void** setText(StringBuilder str) {**text** = str;}  
 **public** StringBuilder getText(){**return text**;}  
 **public void** setPlus(){  
 **int** n;  
 String str = **""**;  
 **for** (**int** i = 0; i < **text**.length() - 2; i++){  
 **if** (**text**.charAt(i) > **'0'** && **text**.charAt(i) <= **'9'** &&  
 **text**.charAt(i + 1) >= **'0'** && **text**.charAt(i + 1) <= **'9'** &&  
 (**text**.charAt(i + 2) < **'0'** || **text**.charAt(i + 2) > **'9'**)){  
 *// В условии проверяется что бы данная ячейка была цифрой, следующая была цифрой, а 3 уже не была* n = Integer.*parseInt*(**text**.substring(i, i+2));  
 **for** (**int** j = 0; j < n; j++)  
 str+= **"+"**;  
 **text** = **text**.replace(i, i + 2, str);  
 str = **""**;  
 }  
 i = **text**.indexOf(**" "**, i + 1);  
 **if** (i < 0) **break**;  
 }  
 }  
 **public void** addKONEC(){  
 StringBuilder buf = **new** StringBuilder(**text**);  
 **int** minLength = buf.length(), minInd = 0, n;  
 **for** (**int** i = 0; i < buf.length(); i++){  
 **if** (buf.charAt(i) == **'.'** || buf.charAt(i) == **'!'** || buf.charAt(i) == **'?'**){  
 buf.replace(i, i+1, **"\n"**);  
 }  
 }  
 **for** (**int** i = 0; i < buf.length(); i++){  
 n = buf.indexOf(**"\n"**, i);  
 **if** (minLength > n - i){  
 minLength = n - i;  
 minInd = n;  
 }  
  
 i = n;  
 }  
 **text**.replace(minInd, minInd + 1, **"КОНЕЦ."**);  
 }  
 **public void** changeNum() {  
 StringBuilder buf;  
 **for** (**int** i = 0; i < **text**.length() - 3; i++) {  
 **if** (**text**.charAt(i) > **'0'** && **text**.charAt(i) <= **'9'** &&  
 **text**.charAt(i + 1) >= **'0'** && **text**.charAt(i + 1) <= **'9'** &&  
 **text**.charAt(i + 2) >= **'0'** && **text**.charAt(i + 2) <= **'9'** &&  
 (**text**.charAt(i + 3) < **'0'** || **text**.charAt(i + 3) > **'9'**)) {  
 buf = **new** StringBuilder(**text**.substring(i, i + 3));  
 **text**.replace(i, i + 3, buf.reverse().toString());  
 }  
 }  
 }  
}