



**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 09.04.01/12 Интеллектуальный анализ больших
данных в системах поддержки принятия решений.**

О Т Ч Е Т

по лабораторной работе № 7

Вариант № 5

Название: Строки и регулярные выражения

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

А.О.Крейденко

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2024

Цель: изучить работу со строками и регулярными выражениями.

Задание 1: в тексте после k-го символа вставить заданную подстроку.

Код класса Main:

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Ввод исходного текста
        System.out.println("Введите исходный текст:");
        String text = scanner.nextLine();

        // Ввод значения k
        System.out.println("Введите позицию k:");
        int k = scanner.nextInt();
        scanner.nextLine(); // потребление новой строки после
nextInt()

        // Ввод подстроки
        System.out.println("Введите подстроку для
вставки:");
        String substring = scanner.nextLine();

        // Проверка допустимости значения k
        if (k < 0 || k >= text.length()) {
            System.out.println("Некорректное значение k.");
            return;
        }

        // Вставка подстроки после k-го символа
        String result = insertSubstring(text, k, substring);

        // Вывод результата
        System.out.println("Результат:");
        System.out.println(result);
    }

    // Метод для вставки подстроки после k-го символа
    public static String insertSubstring(String text, int k,
String substring) {
        String before = text.substring(0, k);
        String after = text.substring(k);
        return before + substring + after;
    }
}
```

Работа программы показана на рисунке 1.

```
Введите исходный текст:
Hello world
Введите позицию k:
5
Введите подстроку для вставки:
hello
Результат:
Hellohello world

Process finished with exit code 0
```

Рисунок 1 – Работа программы 1

Задание 2: после каждого слова текста, заканчивающегося заданной подстрокой, вставить указанное слово.

Код класса Main:

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Ввод исходного текста
        System.out.println("Введите исходный текст:");
        String text = scanner.nextLine();

        // Ввод подстроки
        System.out.println("Введите подстроку для поиска:");
        String substring = scanner.nextLine();

        // Ввод слова для вставки
        System.out.println("Введите слово для вставки:");
        String wordToInsert = scanner.nextLine();

        // Вставка слова после каждого слова, заканчивающегося
        на подстроку
        String result = insertAfterSubstring(text, substring,
wordToInsert);

        // Вывод результата
        System.out.println("Результат:");
        System.out.println(result);
    }

    // Метод для вставки слова после каждого слова,
    заканчивающегося на заданную подстроку
```

```

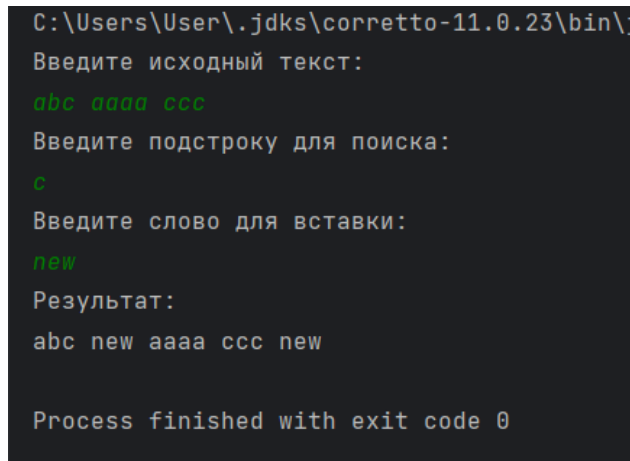
        public static String insertAfterSubstring(String text,
String substring, String wordToInsert) {
    String[] words = text.split("\\s+");
    StringBuilder result = new StringBuilder();

    for (String word : words) {
        result.append(word);
        // не учитываем запятую в конце слова, если она есть
        if (word.replace(",", "", "").endsWith(substring)) {
            result.append(" ").append(wordToInsert);
        }
        result.append(" ");
    }

    // Удаление последнего пробела
    return result.toString().trim();
}
}

```

Работа программы показана на рисунке 2.



```

C:\Users\User\.jdk\corretto-11.0.23\bin\java.exe
Введите исходный текст:
abc aaaa ccc
Введите подстроку для поиска:
c
Введите слово для вставки:
new
Результат:
abc new aaaa ccc new
Process finished with exit code 0

```

Рисунок 2 – Работа программы 2

Задание 3: в стихотворении найти количество слов, начинающихся и заканчивающихся гласной буквой.

Код класса Main:

```

import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Ввод стихотворения
        List<String> poem = new ArrayList<>();
        System.out.println("Введите стихотворение:");
    }
}

```

```

        String line = scanner.nextLine();
        while(!line.isEmpty()) {
            poem.add(line);
            line = scanner.nextLine();
        }
        scanner.nextLine();

        // Подсчет слов, начинающихся и заканчивающихся гласной
буквой
        int count = 0;
        for (String l : poem) {
            count += countVowelWords(l);
        }

        // Вывод результата
        System.out.println("Количество слов, начинающихся и
заканчивающихся гласной буквой: " + count);
    }

    // Метод для подсчета слов, начинающихся и заканчивающихся
гласной буквой
    public static int countVowelWords(String text) {
        String[] words = text.split("\\s+");
        int count = 0;

        for (String word : words) {
            if (isVowelWord(word)) {
                count++;
            }
        }

        return count;
    }

    // Метод для проверки, начинается и заканчивается ли слово
гласной буквой
    public static boolean isVowelWord(String word) {
        if (word.isEmpty()) {
            return false;
        }

        // Преобразуем слово в нижний регистр для упрощения
сравнения
        word = word.toLowerCase();

        char firstChar = word.charAt(0);
        char lastChar = word.charAt(word.length() - 1);

        return isVowel(firstChar) && isVowel(lastChar);
    }

    // Метод для проверки, является ли символ гласной буквой
    public static boolean isVowel(char c) {

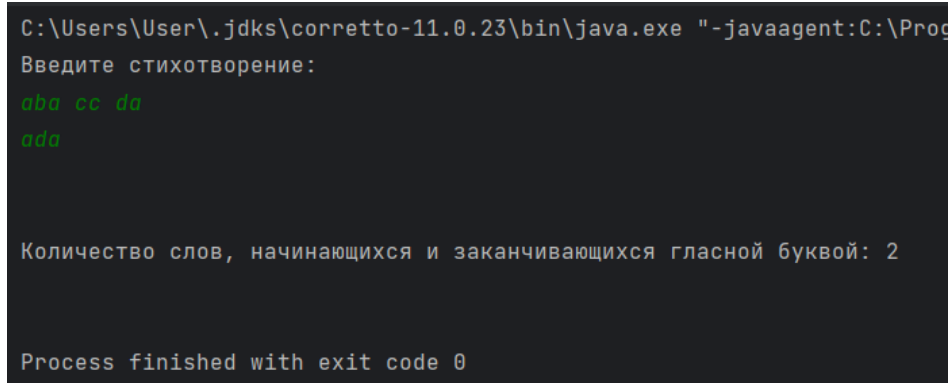
```

```

        return "ауоыиэяюёеаеиуѳ".indexOf(c) != -1;
    }
}

```

Работа программы показана на рисунке 3.



```

C:\Users\User\.jdk\corretto-11.0.23\bin\java.exe -javaagent:C:\Pro...
Введите стихотворение:
aba cc da
ada

Количество слов, начинающихся и заканчивающихся гласной буквой: 2

Process finished with exit code 0

```

Рисунок 3 – Работа программы 3

Задание 4: напечатать без повторения слова текста, у которых первая и последняя буквы совпадают.

Код класса Main:

```

import java.util.HashSet;

public class Main {
    public static void main(String[] args) {
        String text = "Слово, еще одно слово";

        // Разбиваем текст на слова
        String[] words = text.split("\\s+");

        // Создаем множество для хранения уникальных слов,
        // удовлетворяющих условию
        HashSet<String> uniqueWords = new HashSet<>();

        // Перебираем каждое слово из текста
        for (String word : words) {
            // Проверяем условие: первая и последняя буквы
            // совпадают
            if (word.length() > 0 && word.charAt(0) ==
                word.charAt(word.length() - 1)) {
                // Добавляем слово в множество, если оно уникально
                uniqueWords.add(word);
            }
        }

        // Выводим уникальные слова
        System.out.println("Уникальные слова с совпадающими
        первой и последней буквами:");
    }
}

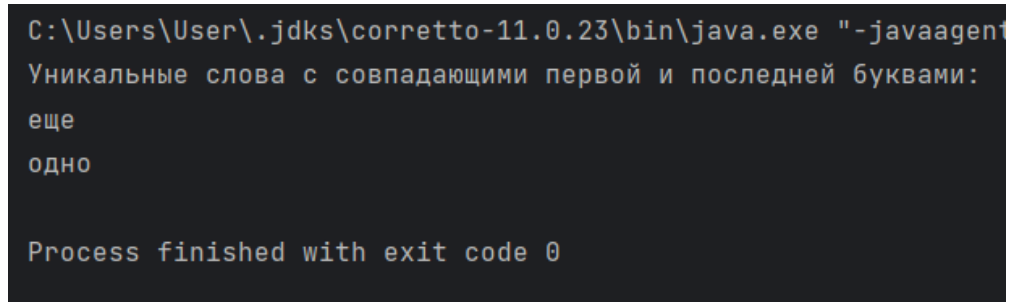
```

```

        for (String uniqueWord : uniqueWords) {
            System.out.println(uniqueWord);
        }
    }
}

```

Работа программы показана на рисунке 4.



```

C:\Users\User\.jdk\corretto-11.0.23\bin\java.exe -javaagent...
Уникальные слова с совпадающими первой и последней буквами:
еще
одно

Process finished with exit code 0

```

Рисунок 4 – Работа программы 4

Задание 5: в каждом предложении текста поменять местами первое слово с последним, не изменяя длины предложения.

Код класса Main:

```

public class Main {
    public static void main(String[] args) {
        // Исходный текст
        String text = "Это пример предложения. Еще одно предложение. И наконец третье предложение.";

        // Разбиваем текст на предложения по точке с пробелом
        String[] sentences = text.split("\\.\\s+");

        // Обходим каждое предложение
        for (int i = 0; i < sentences.length; i++) {
            // Разбиваем предложение на слова по пробелам
            String[] words = sentences[i].split("\\s+");

            // Меняем местами первое и последнее слово в предложении
            String firstWord = words[0];
            words[0] = words[words.length - 1];
            words[words.length - 1] = firstWord;

            // Собираем предложение обратно из слов
            StringBuilder newSentence = new StringBuilder();
            for (String word : words) {
                newSentence.append(word).append(" ");
            }

            // Заменяем исходное предложение на новое в массиве предложений
            sentences[i] = newSentence.toString().trim();
        }
    }
}

```

```

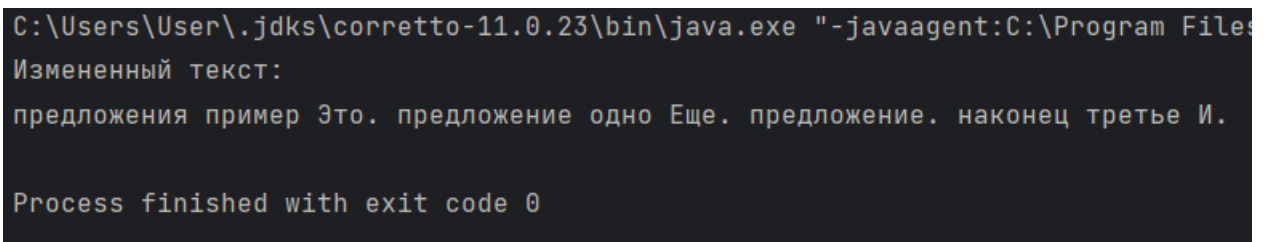
    }

    // Собираем текст обратно из предложений
    StringBuilder newText = new StringBuilder();
    for (String sentence : sentences) {
        newText.append(sentence).append(". ");
    }

    // Выводим измененный текст
    System.out.println("Измененный текст:");
    System.out.println(newText.toString().trim());
}
}

```

Работа программы показана на рисунке 5.



```

C:\Users\User\.jdk\corretto-11.0.23\bin\java.exe -javaagent:C:\Program Files\...
Измененный текст:
предложения пример Это. предложение одно Еще. предложение. наконец третье И.
Process finished with exit code 0

```

Рисунок 5 – Работа программы 5

Задание 6: в предложении из n слов первое слово поставить на место второго, второе – на место третьего, и т.д., $(n-1)$ -е слово – на место n -го, n -е слово поставить на место первого. В исходном и преобразованном предложениях между словами должны быть или один пробел, или знак препинания и один пробел.

Код класса Main:

```

public class Main {
    public static void main(String[] args) {
        // Исходное предложение
        String sentence = "В этом предложении слова меняются местами";
        System.out.println("Исходное предложение:");
        System.out.println(sentence + "\n");

        // Разбиваем предложение на слова
        String[] words = sentence.split("\\s+");

        // Меняем местами слова
        String lastWord = words[words.length - 1];
        for (int i = words.length - 1; i > 0; i--) {
            words[i] = words[i - 1];
        }
    }
}

```



```

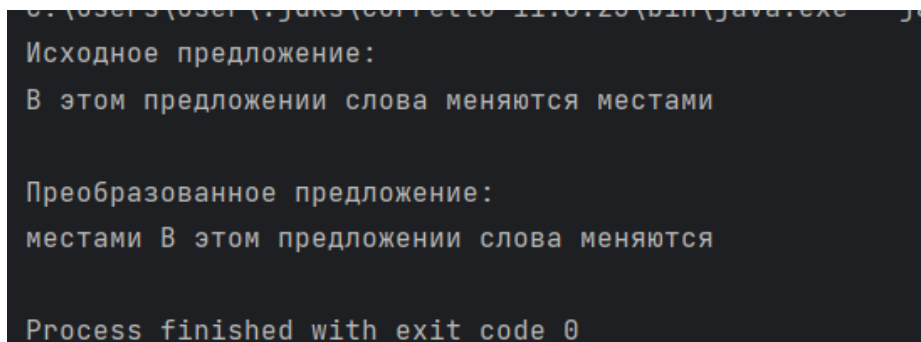
words[0] = lastWord;

// Собираем предложение обратно из слов
StringBuilder newSentence = new StringBuilder();
for (String word : words) {
    newSentence.append(word).append(" ");
}

// Выводим измененное предложение
System.out.println("Преобразованное предложение:");
System.out.println(newSentence.toString().trim());
}
}

```

Работа программы показана на рисунке 6.



```

C:\Users\user>java -cp .\bin\java.exe ...
Исходное предложение:
В этом предложении слова меняются местами

Преобразованное предложение:
местами В этом предложении слова меняются

Process finished with exit code 0

```

Рисунок 6 – Работа программы 6

Задание 7: заменить все одинаковые рядом стоящие символы в тексте одним символом.

Код класса Main:

```

public class Main {
    public static void main(String[] args) {
        // Исходный текст
        String text = "Замммменить все одддинаковые рядоммм  
стоящие символы в тексте одниммм символом.";
        System.out.println("Исходный текст: ");
        System.out.println(text + "\n");

        // Переменная для хранения преобразованного текста
        StringBuilder modifiedText = new StringBuilder();

        // Переменная для хранения последнего уникального символа
        char lastChar = '\0'; // '\0' - это символ конца строки,
        чтобы не совпадал с первым символом

        // Обходим каждый символ в тексте
        for (int i = 0; i < text.length(); i++) {
            char currentChar = text.charAt(i);

```

```

        // Если текущий символ отличается от предыдущего,
        добавляем его в преобразованный текст
        if (currentChar != lastChar) {
            modifiedText.append(currentChar);
            lastChar = currentChar;
        }
    }

    // Выводим преобразованный текст
    System.out.println("Преобразованный текст:");
    System.out.println(modifiedText.toString());
}
}

```

Работа программы показана на рисунке 7.

```

Исходный текст:
Замммменить все оддддинаковые рядоммм стоящие символы в тексте одниммм символом.

Преобразованный текст:
Заменить все одинаковые рядом стоящие символы в тексте одним символом.

Process finished with exit code 0

```

Рисунок 7 – Работа программы 7

Задание 8: вывести в заданном тексте все слова, расположив их в алфавитном порядке.

Код класса Main:

```

import java.util.Arrays;
import java.util.Comparator;

public class Main {
    public static void main(String[] args) {
        // Исходный текст
        String text = "Вывести в заданном тексте все слова,
        расположив их в алфавитном порядке.";

        // Разбиваем текст на слова
        String[] words = text.split("\\s+");

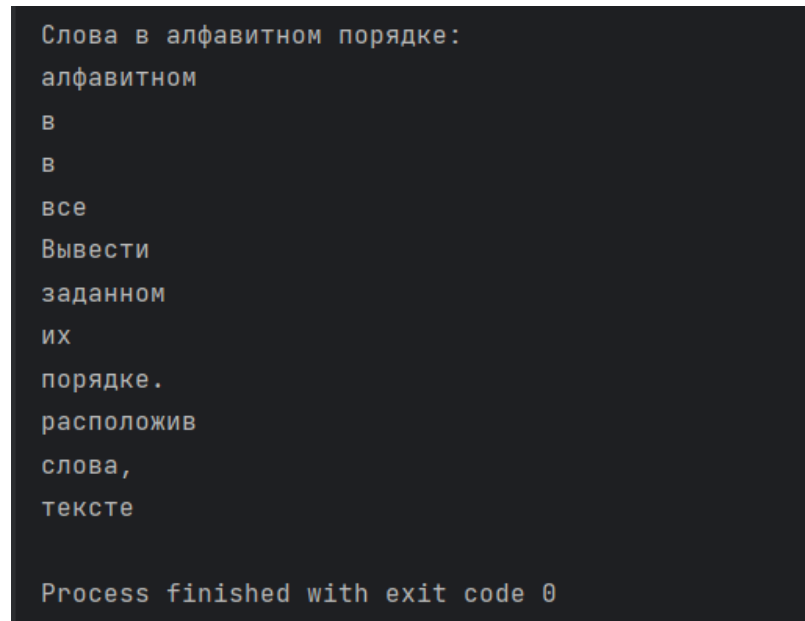
        // Сортируем слова в алфавитном порядке с учетом
        регистра
        Arrays.sort(words,
        Comparator.comparing(String::toLowerCase));

        // Выводим отсортированные слова
        System.out.println("Слова в алфавитном порядке:");
        for (String word : words) {
            System.out.println(word);
        }
    }
}

```

```
}  
}  
}
```

Работа программы показана на рисунке 8.



```
Слова в алфавитном порядке:  
алфавитном  
в  
в  
все  
Вывести  
заданном  
их  
порядке.  
расположив  
слова,  
тексте  
  
Process finished with exit code 0
```

Рисунок 8 – Работа программы 8

Вывод: во время выполнения лабораторной работы была изучена работа со строками и регулярными выражениями.